VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Systém pro sběr, agregaci a vyhodnocování událostí monitorovaných při provozu počítačové sítě

The Systém of Collection, Agregation and Evaluation of Monitored Events on the Computer Network

2012 **Bc. Ondřej Pavlík**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Ondřej Pavlík**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Systém pro sběr, agregaci a vyhodnocování událostí monitorovaných při provozu počítačové sítě
The System of Collection, Aggregation and Evaluation of Monitored Events on the Computer Network

Zásady pro vypracování:

Cílem práce je realizovat systém pro inteligentní agregaci a finální prezentaci událostí produkovaných více různými zdroji při provozu počítačové sítě. Práce bude zahrnovat stručnou analýzu způsobu sběru událostí či abnormalit a dále návrh, implementaci a praktické odzkoušení aplikace, nad testovacími daty.
1. Prostudujte existující mechanizmy a aplikace sloužící ke sběru, shromažďování a agregaci událostí produkovaných prvky počítačové sítě (SNMP,
NetFlow, NetConf, Zabbix, Centreon, Cisco řešení - MARS).
2. Navrhněte vlastní řešení, které bude související události z různých zdrojů agregovat a finálně prezentovat.
3. Navržené řešení implementujte.
4. Implementovanou aplikaci otestujte a zhodnoťte dosažené výsledky.

Seznam doporučené odborné literatury:

HALLEEN, Gary; KELLOGG, Greg. Security Monitoring with Cisco Security MARS. USA : Cisco Press, 2007. 336 s. ISBN 978-1-58705-270-5.
FRY, Chris; NYSTROM, Martin. Security Monitoring. USA : O'Reilly Media, Inc., 2009. 256 s. ISBN 978 -0-596-51816-5.
KRETCHMAR, James M. Open source network administration. USA : Prentice Hall PTR, 2004. 256 s. ISBN 978-0-13-046210-7.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Martin Milata**

Datum zadání: 19.11.2010

Datum odevzdání: 04.05.2012

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: 30.dubna 2012

Ondřej Pavlík

.....

Podpis

Poděkování

Rád bych poděkoval Ing. Martinu Milatovi za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Abstrakt

Zpráva se v první části věnuje možnostem sběru, předávání a vyhodnocení událostí na jednotlivých PC a síťových prvcích. V druhé části, na základě získaných informací z části první, je popsán odzkoušený princip a návrh, jak je možné tyto události agregovat a prezentovat prostřednictvím monitorovacích agentů a centrálního pultu. Zpráva dále popisuje jednotlivé části systému, z kterých byl sestaven.

Klíčová slova

Informační systém, databáze, syslog, log, agregace, událost, monitoring, síť, server, agent,

Abstract

The first section of the Report addresses the alternatives of collecting, transmitting, and evaluating the events on individual PSs, and on the Network. Based on the information obtained from the first section of the Report, the second section illustrates the tested principle (*should say tested principle of what?*) and the proposal of how to best aggregate and present these events through monitoring agents and the logs. In addition, the Report further describes and explains the individual components of the (*what system?*) system.

Key words

Information system, database, syslog, log aggregation, event monitoring, network, server, agent,

Obsah

1 Úvod

Efektivita provozu počítačové sítě úzce souvisí s bezpečností a spolehlivostí. Zajištění spolehlivého a bezpečného provozu sítě proto vyžaduje racionální a efektivní správu této sítě, která je možná pouze s využitím vhodných technologií, které zajištění provozu sítě a jejího řízení zefektivní.

U správců počítačových sítí je třeba myslet na to, že jejich pracovní povinnosti nekončí koncem pracovní doby. Žijeme v době, kdy internet a počítačové sítě spojují svět. Všichni uživatelé spoléhají na stoprocentní spolehlivost dostupnosti počítačových systémů, které dnes hýbou světem.

Kontinuální provoz počítačových sítí vyžaduje také i kontinuální dohled, aby nedošlo k výpadku. A pokud už k němu dojde, je důležité zajistit, aby výpadek byl co nejkratší. Na dohled na dílčích částech internetové sítě není většinou pracovník sám. Zajišťuje dohled pod záštitou a za podpory odborné organizace jako jsou: Telefonica O2, UPC, AutoCont atp. Vzniká tedy možnost zastoupení jednotlivce jiným zaměstnancem a pracovník odborného servisu tak má možnost si odpočinout. Jinak je tomu ovšem u firmy živnostenské, která má jediného pracovníka. Živnostník, který je smluvně sjednán na údržbu několika firemních sítí, není ze strany zákazníků nikterak zohledňován, jeho povinnosti jsou tytéž, jako u velkých organizací, tj. zajistit kontinuální bezporuchový provoz sítě. Nikdo nebere ohled na to, zda má dotyčný dodavatel po pracovní době nebo ne. V případě poruchy bude dodavatel volán k havárii třeba i o půlnoci. V řadě případů není v moci zákazníků havárii odstranit a stejně často se havárie objevují ve chvíli, kdy uživatel sítě pracuje na urgentní zakázce a výpadek pro něj může mít fatální následky.

Z toho všeho co je zde uvedeno, plyne jediné. Bez ohledu na to, zda správu sítě vykonává velká organizace, nebo živnostník je třeba, aby správce sítě měl neustálý přehled o tom co se děje na síti a počítačích svěřených do jeho péče. Není ale v možnostech správců být u všech prvků současně a současně na ně dohlížet.

Z ekonomických důvodů se zákazníci obracejí většinou na řešení, které jim je nabízeno jako nejlevnější. Bez specializovaného, zpravidla nákladného software pak ale vzniká pro správce často nepříjemný problém. Jak v co nejkratším možném čase, který právě je tak drahý, mít přehled o všech prvcích na síti, tak aby to bylo co nejekonomičtější a současně zákazníkům nabídli co nejlepší službu.

Není možné tedy správcům sítí přidávat rutinní prácí, kterou by místo nich mohly řešit inteligentní aplikace a sofistikované systémy.

Výrobci hardwaru už přišli na to, že je třeba dávat informace o stavu svých zařízení k dispozici administrátorů v nějaké přehledné podobě. Proto už existují mnohá zařízení, která jsou schopna zaznamenávat svá chování do souborů (zápis logů) pro případné zásahy správců.

Není ale v silách většiny výrobců, aby dodali dostatečně sofistikovaný a přitom dostatečně levný systém, který by byl schopen vyhodnocovat dění na všech síťových prvcích. Většinou je důvodem

specializace firmy na jistý HW, jenž je pouze částí síťového celku. Pokud se zákazníkovi podaří najít firmu, která takové řešení nabízí (je jich opravdu málo), pak bývá řešení velmi nákladné.

Externí firma většinou navíc je zpravidla postavena před nepříjemnou skutečnost. Zákazník jí představí svou hotovou počítačovou síť, jejíž správu požaduje. Často se však jedná o nesourodou a často nekoncepčně budovanou síť různorodého hardware. Ten není možné centrálně a tedy ani efektivně spravovat. Správci systému nezbývá nic jiného, než navýšit cenu za správu, nebo najít řešení, jak takovou heterogenní síť transformovat do jediného velkého udržovatelného celku.

Jsem jedním z takových správců, který řeší problém nesourodosti hardware v nekoncepčně budovaných sítích. Mám na starosti údržbu celé sítě v jedné Základní škole na Jesenicku a ještě v pár menších firmách. Protože jsem na to sám a mou povinností je udržet aktualizované, zabezpečené a funkční všechny počítače ve všech sítích do mé péče svěřené hledal jsem nejoptimálnější způsob, jak toho dosáhnout, aniž bych tím nebyl příliš časově vytížen. Správa sítě není jedinou mou aktivitou a svým časem se snažím šetřit, kde se dá. Ten, kdo zkusil pracovat během studia, ten jistě chápe moje slova. Je zřejmé, že při studiu a uvedeném pracovním vytížení mne výše popsaný problém eminentně zajímá, a proto jsem si jej zvolil jako téma mé diplomové práce.

Realizovat systém pro inteligentní agregaci a finální prezentaci událostí produkovaných více různými zdroji při provozu počítačové sítě.

Co vlastně znamená pojem *agregace*? Výraz pochází z latinských slov „*ad*“ ve významu „*k*“ nebo předpony „*při-*“ a „*grex*“ znamenajícím „*stádo*“. Tedy v doslovném znění „*připojení ke stádu*“. I v dnešním moderním pojetí nese výraz „*agregace*“ význam *připojování*, *seskupování*, *shlukování*, či *shrnutí*. V našem konkrétním případě seskupení informací z různých zdrojů do jediného místa, kde mohou být zejména ty nejdůležitější informace přehledně prezentovány tak, aby správce byl včas upozorněn na případné provozní problémy. Cíl této práce by tedy ve svém důsledku měl vést k řešení, které bude šetřit čas při správě sítí a navíc takovým způsobem aby správce o vzniklých problémech věděl dříve, než zákazník.

Na základě konzultace s garantem jsem došel k závěru, že pokud má mít správce přehled o všech prvcích umístěných na síti, musí mít k dispozici systém, který bude mít za cíl sesbírat maximální množství informací o jednotlivých prvcích sítě, tedy počítačích, přepínačích, směrovačích, serverech, atp. Tyto informace pak bude nutné agregovat pomocí nějakých pravidel, tak aby správce sítě nebyl zahlcen velkým množstvím informací, které na funkčnost systému nemají větší vliv.

Z tohoto hlediska tedy úkol sestává ze dvou základních problémů:

- sběr informací o síťových prvcích
- účelové filtrování a agregace informací

Z výše uvedených skupin problémů vyplývá rozsáhlý řešitelský úkol. Prozkoumání značného množství komunikačních protokolů a také aplikací pro sběr dat, které už jsou na trhu dostupné. Z toho všeho je nutno vybrat podstatné a pro tuto práci použitelné principy, jež by umožnily navrhnout takový systém, který nebude pro většinu uživatelů finančně nedostupný, a přitom bude přinášet informace podstatné pro správce, přičemž tento systém bude možno nasadit v jakémkoli síťovém prostředí. Myšleno prostředí operačních systémů a software, instalovaném na síťových prvcích.

Následující kapitoly popisují způsob řešení a základní principy, které jsem při zpracovávání problému prostudoval a použil, příp. nepoužil z důvodů, které uvádím. V první části práce popisuji jednotlivé studované mechanizmy, protokoly, systémy a aplikace. Ve druhé části diplomové práce se, na základě získaných informací, snažím navrhnou, popsat a naimplementovat systém, který by vyhovoval výše uvedeným potřebám a parametrům.

2 Události v síti a možné problémy

Na monitoring událostí se můžeme dívat ze dvou pohledů. Chceme se dozvědět, že někde došlo k problému. Tedy, že něco přestalo fungovat či byl překročen nějaký kritický limit. Nebo chceme získávat aktuální (ale i historické) informace o určitém systému. To může být pohled na vytížení serveru, abychom plánovali jeho další využití. Přehled, kde v síti (do jakého portu jakého přepínače) je připojen klient s jakou IP a MAC adresou. Či sledování vytížení datových linek.

2.1 Oblasti pro monitoring událostí

Z obecnějšího hlediska můžeme uvést monitoring událostí:

- serverů a jejich služeb
- aktivních síťových prvků
- síťové komunikace/provozu
- bezpečnosti

K tomu se přidávají určité specifické oblasti, kdy si v základu vystačíme se stejným monitoringem událostí, jako pro výše uvedené skupiny, ale můžeme získat více použitím specializovaného nástroje. To jsou oblasti jako IP telefonie, bezdrátové sítě a virtuální prostředí.

- Když půjdeme hlouběji, tak nás může zajímat:
- dostupnost serverů
- dostupnost služeb/aplikací (spolu s latencí – reakční dobou)
- události na serverech
- vytížení zdrojů (procesor, paměť, disk)
- vytížení linek – měření přenosu dat
- statistiky síťového provozu
- analýzy nestandardního chování v síti
- informace o portech přepínačů
- monitoring speciálních oblastí jako je WiFi či IP telefonie
- bezpečnostní incidenty

2.2 Technologie pro monitoring událostí

Pokud použijeme nějaký komplexní monitorovací systém pro dohled serverů, tak máme většinou dvě základní možnosti pro přístup k informacím.

Sledovat události s agentem, kdy na server instalujeme speciálního klienta. Musíme tedy mít k dispozici agenta pro daný operační systém, musíme mít možnost na sever instalovat a přibývá další

aplikace, která může způsobovat problémy. Na druhou stranu dostaneme širokou škálu údajů, které můžeme ze serveru získávat.

Druhou možností je sledování událostí bez agenta, kdy se testují vlastní služby serveru nebo se data získávají pomocí určitých standardních protokolů a programů (jako PING, WMI, IPMI).

Seznam technologií, které můžeme pro tento monitoring použít, ať již samostatně nebo uvnitř monitorovacího systému.

- dostupnost serveru pomocí ping testu
- dostupnost služby pomocí navázání TCP spojení nebo na aplikační úrovni
- události ze serverů – Systémový log (Syslog)
- získávání údajů pomocí klienta
- získávání údajů pomocí monitorovacích protokolů WMI, SNMP, IPMI
- sledování síťových toků - NetFlow
- analýza síťových protokolů - network protocol analyzer
- bezpečnost v síti - IDS/IPS

2.3 Zaznamenávání událostí – reporty a alerty

Můžeme si nastavit perfektní sledování událostí, které bude hlídat a zaznamenávat vše v naší síti, ale pokud nebudeme mít přehledné, dostupné a často i inteligentní výstupy, tak ničeho nedosáhneme. Proto je důležité již od začátku plánovat, jaký výstup pro jakou oblast je pro nás nejvhodnější.

Z pohledu typů dat máme dvě základní oblasti. První jsou události, které získáme pomocí Systémového logu, WMI či SNMP trapů ze serverů, přepínačů a dalších zařízení. Druhou jsou hodnoty, často číselné, ukazující okamžitý stav nějaké vlastnosti získané třeba programem PING. K tomu se ještě přidává otázka, zda nás zajímá okamžitý stav nebo potřebujeme ukládat historii, jak se hodnoty mění v čase.

Pro různé sledované údaje se samozřejmě hodí různá reprezentace. Vytížení procesoru nás zajímá v určité časové periodě a vhodné zobrazení je grafem. Stavy portů přepínače naopak chceme vidět aktuální a přehledná pro tento monitoring může být třeba tabulka. Dostupnost serveru za období se může zobrazit jednou procentuální hodnotou.

Pro globální pohled na síť je výhodná grafická reprezentace, kdy vidíme pohled na schéma sítě nebo její části. Pokud dojde někde k problému, tak se daný prvek zvýrazní a po rozkliknutí dostaneme detailní informace. Vedle toho můžeme mít zobrazeny kategorie událostí podle závažnosti zobrazující počet nevyřešených incidentů.

Výše popsané reprezentace jsou zajímavé a v určitých situacích užitečné. Pokud se však jedná o bezpečnostní nebo havarijní události, a my nemáme dohledový tým, který neustále sleduje centrální pult (dashboard) monitorovacího systému, tak je mnohem užitečnější vygenerování a odeslání emailové nebo SMS zprávy (třeba pomocí IP GSM brány). Můžeme takto zasílat důležité zprávy pro informaci a kritické zprávy, abychom provedli rychlou reakci.

2.4 Dostupnost zařízení/služby

Asi první věcí, kterou začneme monitorovat, je dostupnost serveru či aktivního síťového prvku. V malé firmě, kde máme jeden dva servery, se nedostupnost projeví patrně velice záhy. Ve větším prostředí se však mohou začít kupit problémy navazujících služeb, a může trvat delší dobu, než někdo zaregistruje nedostupnost nějakého serveru či síťové cesty (které bývají často redundantní).

Běžně se dostupnost zařízení zjišťuje pomocí jednoduché metody ICMP echo request/response (tedy ping). Případně se používá tzv. „SNMP ping“, což je SNMP dotaz na běžné OID (object identifier – jednoznačný identifikátor objektu, zařízení). Takto můžeme zjistit, zda je dané zařízení „živé“ a měřit dobu odezvy (latency). Nedozvíme se však, že třeba webový server Apache neběží. Proto je dalším krokem monitoring dostupnosti aplikační služby.

Většina běžných síťových služeb využívá protokol TCP a poslouchá na nějakém portu. Takže můžeme testovat, zda se nám na server podaří navázat TCP spojení na daný port. To znamená, že daná služba běží a poslouchá. Lepším ověřením je aplikační test, kdy ověříme, že se služba chová jak má. Takže například pro webserver se připojíme ke stránce a ověříme, zda vrací v hlavičce kód 200. Pro mail server zavoláme základní SMTP příkazy pro navázání spojení se serverem apod.

3 Mechanizmy ke sběru událostí a komunikaci mezi síťovými prvky

3.1 SNMP

SNMP (Simple Network Managment Protocol) je jednoduchý a velmi často používaný protokol pro získávání nebo nastavování hodnot na síťových prvcích. Paralelou k tomuto protokolu je například WMI (Windows Managment Instrumentation) vytvořený společností Microsoft. SNMP je ale protokol, který je podporován většinou zařízení, na rozdíl od výše zmiňovaného WMI. Jeho podporu můžeme čekat u typu zařízení, jako jsou aktivní síťové prvky (routery, switche, wifi AP), osobní počítače, servery, tiskárny a další zařízení která mohou komunikovat prostřednictvím sítě.

Protokol SNMP začal vznikat v roce 1988 jako reakce na potřebu efektivní platformy pro správy počítačových sítí. V roce 1990 byl institucí IAB (Internet Acitivities Board) potvrzen jako standard sítě internet. První specifikace protokolu, RFC1157, vznikla v roce 1989 a stanovovala vlastnosti SNMP verze 1. Tato specifikace byla pro začátek ideální. Měla příkazy **get**, **get next**, **set** a **trap**, Nebyla pořádně zabezpečena, protože využívala ochranu pouze nešifrovaným heslem nazvaným jako **community string**. Toto byl pravděpodobně hlavní nedostatek, který přetrval ještě ve verzi 2. Ochrana heslem je nedostatečná, protože heslo lze poměrně jednoduše zjistit analyzátorem paketů. Nejnovější definice protokolu SNMP verze 3 je z roku 1998, a umožňuje ochranu přenášených dat pomocí DES algoritmu.

3.1.1 Funkce SNMP

Protokol SNMP je obousměrný komunikační nástroj. Komunikuje z jedné strany v roli správce (manager) a z druhé strany v roli agenta. SMTP pracuje ve dvou režimech:

- ***Správce posílá dotazy agentovi a očekává odpověď*.** Hodnoty proto může získávat i více správců současně. Svůj dotaz mohou na agenta poslat v jakoukoli chvíli.
- ***Agent posílá automaticky oznámení (trapy - oznámení o tom, že se s agentem něco děje, jinými slovy žádost o podrobnější zjištění a nápravu) správci.*** To, kdy se oznámení odešle, záleží na nadefinování agenta. Trapy se mohou odesílat buďto v pravidelných časových intervalech, nebo po překročení nějaké hodnoty. Tyto trapy agent může posílat většinou jen jednomu správci.

  V současné době máme k dispozici tři verze tohoto protokolu.

- ***SNMP verze 1*** byla vytvořena v roce 1988. Podporuje příkazy GET, GET NEXT, SET a TRAP. Největším problémem je slabé zabezpečeni. Hesla se ukládají pomoci takzvaného "community string" nebo-li společného řetězce a jsou uložena a přenášena v nezašifrované podobě, což je zřejmý bezpečnostní nedostatek.

- ***SNMP verze 2c*** – v této verzi byla implementována kontrola doručení, takže ke ztrátě trapů by nemělo dojít. Heslo je stále ve formátu stringu, čili řetězce, a přenášeno opět nezašifrovanou podobou. Čili ani tento protokol nelze považovat za bezpečný.
- ***SNMP verze 3*** byla prohlášena za standart v roce 2004, starší verze jsou IETF prohlášeny jako zastaralé nebo historické. Tato verze konečně umožňuje uspokojivou autentifikaci, a šifrování pomoci DES/AES

3.1.2 SNMP paket

Dotaz a odpověď

Verze	community string	PDU typ	ID dotazu	error status	error ID	OID	hodnota

Příklad

1	public	GET (0)	8	no error (0)	0	1.3.6.1.4.1.311.1.1.3.1.1.1	NULL

Paket trapu

Verze	Community string	PDU typ	enterprise	agent IP	gen trap	spec trap	čas	objekt 1 hodnota 1	…

Community string je heslem pro SNMPv1 a SNMPv2c, *PDU typ* je typ SNMP dotazu.

3.1.3 MIB databáze – Managment Information Base

MIB (Management Information Base) je databáze, která dovoluje jednoznačně identifikovat hodnoty (nastavení), pro konfiguraci zařízení. K tomu, aby mohl SNMP manager i agent tyto informace získat a předávat, je nutná znalost struktury MIB.

Báze dat je objektově orientovaná. Data jsou uložena jako objekty a sdružují se do tříd. Jednotlivé objekty agregují hodnoty. Každý řízený objekt v MIB obsahuje veškeré informace potřebné pro popis. Způsob pojmenování objektů je založen na jejich vztahu. Jeden objekt může obsahovat jiné objekty nebo jiné třídy. MIB je tedy tvořena jedním stromem.

Každý agent by měl udržovat objekty standardní MIB (např. síťové adresy, typy rozhraní, čítače). Jsou definovány tři mechanismy pro přidání:

- přidání nových objektů prostřednictvím definice nové verze MIB-II;
- přidání nestandardních objektů přidáním experimentální větve;
- přidání vlastních objektů v rámci podstromu soukromé větve.

Do MIB byly zařazeny jen nejnutnější objekty. Předem byly vyloučeny objekty svým způsobem nadbytečné, např. ty, které mají konkrétní (např. aritmetické) vztahy s jinými objekty v MIB. Jednoduchost definice a omezená velikost báze umožňuje zaručit minimální dopad na činnost a složitost agentů. To se pak samozřejmě projeví v nárocích na zpracovatelský systém.

3.2 NetFlow

NetFlow je protokol od společnosti Cisco Systems. Jedná se o protokol otevřený. Technologie slouží pro monitorování síťového provozu. Díky tomu jsou schopni administrátoři sítí, popřípadě provozovatelé ISP, řídit toky nejen datové ale i finanční. V současné době pomocí tohoto protokolu ISP řídí cenovou politiku svých nabízených služeb a administrátoři mohou díky těmto statistikám zdokonalovat síť tak aby nikde nedocházelo k většímu přetížení síťových prvků.

3.2.1 Funkce NetFlow

Základní a hlavní funkcí protokolu netflow je sbírání informací o množství přenesených dat. Data se sbírají v uživatelem stanovených bodech na síti, tak aby přinášela maximální informační hodnotu.

3.2.2 Architektura protokolu

Architektura celé služby se většinou skládá z X zařízení tvořících exportéry dat (neboli sondy) a jednoho bodu, do kterého se tato data sbírají, a v kterém se zpracovávají - tzv. kolektor. Exportér je připojen k monitorované lince a analyzuje procházející pakety. Pomocí zachycených toků generuje

statistická data - statistiky. Ty se pak shromažďují na kolektoru. Kolektor je zařízení s velkou úložnou kapacitou obvykle nějaký server, který sbírá statistiky (resp. statistická data) z většího počtu exportérů. Ty ukládá na svých diskových kapacitách. Tato data se dále vyhodnocují a prostřednictvím různých softwarových nástrojů se z nich vytvářejí souhrnné přehledy.

3.2.3 Program Ntop pro monitorování dat z protokolu NetFlow

Programu Ntop pro monitorování dat slouží k přehlednému nahlížení do sesbíraných dat. Instalaci je možné provést podle návodu, který dodává distributor společně s programem. Tento program a zároveň protokol jsem otestoval v laboratořích během předmětu SPS, kde jsem na toto téma vypracoval zprávu s názvem: „Ověření technologie NetFlow na platformě Cisco a Traffic-Flow na platformě Mikrotik“. Tuto zprávu najdete v přílohách diplomové práce jako přílohu B.

3.3 NetConf

Protokol NetConf je určen pro konfiguraci síťových zařízení. Poskytuje jednoduchý mechanismus, pomocí kterého lze provádět nejrůznější změny konfigurace síťového zařízení. Jde nejen o nastavení různých parametrů daného zařízení nebo získání konkrétních konfiguračních či stavových informací, ale hlavě o komplexní změny v konfiguracích síťových zařízení.

Vznik nového protokolu, který sjednotí přístup ke konfiguraci sítí a jednotlivých síťových zařízení, iniciovala na počátku roku 2003 spolu se síťovými administrátory organizace IETF. Nový protokol má postupně nahradit SNMP, který se stal nástrojem spíše pro monitorování sítí, než pro jejich konfiguraci. Síťoví administrátoři viděli příčinu požadované změny hlavně ve velké komplexnosti a s ní spojené složitosti SNMP. To přinášelo problémy hlavně při správě složitějších zařízení. U takových zařízení byl často také problém získat kompletní konfigurační data, která byla reprezentována velkým množstvím objektů. To vadilo zejména administrátorům, protože například nebyli schopni jednoduše porovnat stará konfigurační data s novými. Díky složitosti protokolu SNMP někteří výrobci navíc ani neměli zájem implementovat ve svých zařízení tento protokol. Místo toho vytvářeli vlastní nástroje, které používaly specifický komunikační protokol. Zařízení pak nebylo možné konfigurovat jinými nástroji používajícími SNMP.

Architektura Netconf má čtyři vrstvy. Díky použití XML formátu protokol NetConf zachovává komunikační zprávy relativně jednoduché a čitelné i pro administrátora. Navíc vývoj protokolu NetConf podporovali již od počátku i výrobci síťových zařízení v čele se společnostmi Cisco a Juniper, jejichž některé produkty již NetConf podporují. Po dokončení finální podoby protokolu, která byla zveřejněna v prosinci 2006 jako RFC 4741, se tedy zdá, že protokol bude mít šanci dostát všem očekáváním.

4 Syslog a logy v síti

4.1 Logový soubor

Logový soubor neboli log, někdy též nazýván jako žurnál, si vytvářejí počítačové aplikace a služby, které do tohoto souboru zapisují informace o svojí činnosti. Je určen ke zpětné kontrole pro účely správy. V souboru většinou najdeme informace o tom, kdy daná aplikace byla spuštěna, vypnuta, kdy a kdo do dané aplikace přistupoval. Další informace, které se ukládají do logových souborů jsou výhradně v rukou programátorů dané aplikace.

Existují aplikace, které logování souborů dokážou buď vypnout nebo zapnout. Vše záleží na nastavení. Každá logovaná zpráva má vždy následující atributy:

- prioritu,
- kategorii,
- čas,
- vlastní obsah zprávy.

4.2 Logovací politika v síti

Logovací politika na síti může být pro každou společnost individuální. Mezi obvyklá schémata patří:

1. Zahodit okamžitě veškerá data
2. Periodicky logy mazat
3. Rotovat a po určitou pevnou dobu je skladovat
4. Komprimovat a archivovat logy na nějaké trvalé médium.

Vhodné schéma pro každou organizaci bude jiné, podle množství diskového prostoru a důležitosti bezpečnosti v síti. I když bude mít organizace dostatek prostoru, časem bude muset řešit rakovinné bujení logů.

4.3 Logové zprávy v Linuxových systémech

V operačním systému Linux je pro logové zprávy vyhrazen adresářový prostor /var/log/, do kterého se zapisují logy ze všech běžících služeb a aplikací. Některé služby mají společný logový soubor, jiné si vytváří své vlastní logové soubory. Zpravidla ten nejdůležitější systémový log je označován názvem syslog.

V Linuxových systémech je k dispozici 9 úrovní priority. Jsou jimi (seřazeno od nejnižší priority k nejvyšší):debug, info, notice, warning, err, crit, alert a emerg. Z důvodu zpětné kompatibility

existují ještě synonyma warn (= warning), error (= err) a panic (= emerg), tyto by se ale již neměly používat.

Atribut kategorie říká, jaké oblasti se zpráva týká nebo od jaké služby pochází. K dispozici je 12 předdefinovaných kategorií:

- auth - autentizace, např. zprávy týkající se přihlašování / odhlašování uživatelů,
- authpriv - autentizace, vyhrazeno pro zprávy, které by z bezp. důvodů měly být odděleny od ostatních a které jsou určeny pouze administrátorovi systému,
- cron - zprávy od cronu - démona, který zajišťuje pravidelné spouštění akcí,
- daemon - blíže neurčené zprávy systémových aplikací,
- kern - zprávy jádra,
- lpr - zprávy týkající se tiskového systému (např. lpd apod.),
- mail - zprávy MTA (doručování pošty - např. sendmail apod.),
- mark - vyhrazeno pro tzv. "timestamps"; značky, které se periodicky zapisují do logu,
- news - zprávy NNTP serveru (diskusní skupiny - Use net news),
- security - znamená totéž co "auth", synonymum,
- syslog - zprávy syslogu,
- user - blíže neurčené zprávy uživ. aplikací,
- uucp - zprávy aplikací UUCP (Unix to Unix Copy Protocol, dnes se již téměř nepoužívá).

Existuje ještě 8 dalších kategorií určených k libovolnému použití. Tyto jsou označeny local0 až local7.

4.4 Logové zprávy v systémech Windows

V systémech Windows je logování aplikací a systému řešeno poněkud jinak než je tomu v systémech Linux. Pro ukládání zpráv z aplikací a systému jsou definovány takzvané protokoly událostí, do kterých systém zapisuje zprávy. Protokoly událostí můžeme rozdělit na dvě velké skupiny, a to:

- protokoly systému Windows
- protokoly aplikací a služeb

Kategorie protokolů systému Windows obsahuje protokoly, které jsou dostupné v předchozích verzích systému Windows: protokol aplikací, protokol zabezpečení a systémový protokol. Zahrnuje také dva nové protokoly: protokol instalačního programu a protokol předaných událostí. Protokoly systému Windows jsou určeny k ukládání událostí ze starších aplikací a událostí, které se vztahují k celému systému.

Seznam protokolů je následující:

- Protokol aplikací - obsahuje události zapsané aplikacemi nebo programy
- Protokol zabezpečení - obsahuje události zahrnující platné a neplatné pokusy o přihlášení a také události související s využitím prostředků, jako je vytvoření, otevření a odstranění souborů nebo jiných objektů.
- Protokol instalačního programu - obsahuje události vztahující se k instalaci aplikace.
- Protokol předaných událostí – používá se k uložení událostí sebraných ze vzdálených počítačů.
- Systémový protokol - obsahuje události zaznamenané součástmi systému Windows. Například zaznamenána chyba ovladače nebo jiné systémové součásti, ke které dojde během spuštění.
- Správce – události, které jsou zaměřeny především na koncové uživatele, správce a pracovníky odborné pomoci
- Funkční - události se užívají k analýze a diagnóze problémů nebo podobných případů.
- Analytické - popisují chod programu a označují problémy, které uživatel nemůže sám řešit.
- Ladění - využívají vývojáři při řešení problémů s programy.

Windowsové systémy kategorizují prioritu událostí jiným způsobem než OS Linux. Zprávy mají rozděleny pouze do šesti úrovní. Je jich méně než u operačního systému Linux a jsou jimi:

- *Informace (information)* - signalizuje, že došlo ke změně aplikace nebo součásti.
- *Upozornění (warning)* - označuje, že vznikl problém, který může ovlivnit službu nebo výsledek závažnějším způsobem.
- *Chyba (error)* - znamená, že nastal problém, který může ovlivnit funkci mimo rámec aplikace nebo součásti, která vyvolala danou událost.
- *Kritická (critical)* - označuje, že došlo k chybě, ze které se aplikace nebo součást, jež danou událost spustila, nemůže automaticky zotavit.

Poslední dvě úrovně jsou spíše o právech a přístupech uživatelů. Protože zobrazují informace o zabezpečení.

- *Audit úspěšných provedení operací* - úroveň označuje, že vykonání uživatelských práv bylo úspěšné.
- *Audit neúspěšných provedení operací* - úroveň označuje, že vykonání uživatelských práv bylo neúspěšné.

Protože jsem se ve své diplomové práci vyvíjel aplikaci právě pro sbírání logových záznamů operačního systému Windows, věnuji jim větší pozornost a popíšu, z čeho je složena taková událost, která je zaznamenána do protokolu.

Samotný záznam události v OS MS Windows v sobě nese následující informace:

- Zdroj události
- ID události
- Úroveň
- Uživatel
- Kód operace
- Protokol
- Kategorie úlohy
- Klíčová slova
- Počítač
- Datum a čas

Velkou výhodou zpráv v operačních systémech Windows je, že celá struktura logování událostí je od operačních systémů verze Vista vedena ve formátu XML. Proto je jednodušší a rychlejší vyhledávání záznamů, seskupování podle různých filtrů a možnost exportu. Export se provádí pomocí integrované aplikace do systému nesoucí název *„Prohlížeč událostí“*. Samozřejmě je možné s logovými informacemi pracovat stejně tak, jak s nimi pracuji já pomocí vlastní aplikace. Jedinou podmínkou pro správný chod je potřeba zajistit, aby aplikace běžela ve správcovském módu. Pojem správcovský mód znají až novější typy operačních systémů (Windows Vista a Windows 7).

4.5 Syslog – události ze serverů

Protokol syslog je standard pro přeposílání zpráv z logů po síti. Slouží k tomu, abychom koncentrovali logy z různých zařízení a jejich aplikací na jedno místo a mohli na ně reagovat. Na klientovi potřebujeme aplikaci, která odesílá zprávy z logu, tak jak přibývají, pomocí protokolu Syslog. A potom potřebujeme Syslog server, který tyto zprávy přijímá a zpracovává.

V oblasti Linuxu se jedná o běžnou záležitost. Trochu složitější je situace v *„Microsoftím“* světě. OS Windows si sami vytváří řadu logů označovaných jako Event Log a různé servery přidávají další logy. Tyto logy jsou v MS formátu jak již bylo zmíněno výše a nativně nemají žádnou podporu pro Syslog. Na internetu však nalezneme zdarma aplikace, které fungují jako Syslog server, například Kiwi Syslog Server. A také klienty, kteří přeposílají vybrané události z Event Logu, příkladem je Snare Agent for Windows nebo SaberNet NTsyslog.

Syslog je velice užitečný, protože do logů přibývají běžně stovky zpráv za minutu a pro desítky zařízení nemáme šanci tyto informace procházet. Na Syslogu můžeme vytvořit skripty, které analyzují příchozí zprávy a upozorní nás na problémy. Například ze Security logu Windows můžeme číst špatné pokusy o přihlášení a když se jeden účet snaží v určitém časovém intervalu přihlásit mnohokrát, tak odeslat email správci, že se může jednat o útok.

Druhou výhodou je, že můžeme uchovávat velké množství zpráv, tedy dlouhou historii. Řada zařízení dokáže lokálně uložit pouze omezené množství logů. Navíc máme tyto logy dostupné, i když není dostupný server. Pokud došlo na serveru k poruše nebo byl napaden útočníkem a my se nemůžeme přihlásit, abychom si přečetli lokální log. Tak na Syslogu nalezneme zprávy, které se uložily těsně před tím, než přestal server odpovídat.

5 Systémy a aplikace ke sběru, shromažďování a agregaci událostí

5.1 IDS - Intrusion Detection Systém (systém pro odhalení průniku)

V informatice je pod touto zkratkou nazýván obranný systém počítačových sítí, který monitoruje provoz na síti a snaží se zjistit podezřelé nebo nebezpečné aktivity. Nejdůležitější činností IDS systémů je tedy právě detekce neobvyklých aktivit, které by mohly znamenat poškození monitorované sítě nebo operačního systému. Cílem systému není jen detekovat pokusy o poškození monitorovaného prostředí, ale i detekování činností, které útokům předchází. Mezi takové akce patří například sbírání informací potřebných k útoku skenování otevřených i neotevřených portů na firewallu atd. Hlavním prvkem systému je senzor, který je schopen výše popsanou aktivitu vykonávat.

5.1.1 Typy IDS

1. HIDS – (Host-based intrusion detection systém) – typ, kdy v zařízení je instalován softwarový klient, který se snaží detekovat potenciální útoky systémových volání, činností aplikací, nebo úprav na souborových systémech.
2. NIDS – (Network intrusion detection systém) – jde o nezávislou platformu, která zkoumá síťovou komunikaci a monitoruje dění na síťových prvcích. Systém bývá instalován na zařízeních, přes které protéká síťový provoz (např.: HUB, switch, router). Jsou nakonfigurovány tak aby zrcadlily veškerý provoz na portech a ten pak analyzovaly. V analyzovaných paketech systém hledá kusy škodlivého kódu, který by mohl zničit nebo oslabit funkce monitorované sítě či stanic.

5.1.2 Pasivní nebo aktivní IDS?

Při výběru lze určitě doporučit volbu aktivního systému. Proč?

Na rozdíl od aktivního systému je pasivní systém opravdu pasivním ve smyslu obrany před nežádoucími útoky. Jeho cílem je pouze monitorovat a popřípadě upozornit správce sítě (či systému) na patřičný nalezený problém formou zprávy.

Aktivní systém má možností mnohem víc. Na základě zjištěného útoku aktivní systém vyhodnotí problém a daný problém sám vyřeší (je-li to možné) a správci sítě (systému) pak přijde jen informativní zpráva, že vznikl problém, který byl řešen patřičným způsobem. Aktivní systémy si jsou schopny sami přeprogramovat firewally tak, aby zabránily průniku škodlivého kódu do klientských stanic, nebo mohou přesměrovat aktivitu jinam, tak aby bylo možné zachytit činnost, kterou chce útočník oslabit systém a přitom nedošlo k poškození stanic na které je útočeno.

Za aktivní systém, ale nemůžeme považovat například antivirový program, firewall nebo nástroje pro opravu chyb systému. Aktivním systémem můžeme nazvat IPS (Instrusion prevention systems) známý též pod názvem IDPS (instrusion detection and preventiv systém).

5.2 IPS - Intrusion prevention systems

Jak už bylo uvedeno v předchozí kapitole, intrusion prevention systems, jsou systémy, které monitorují síťovou infrastrukturu a aktivně se podílejí na minimalizaci průniku škodlivých kódů a hackerů do monitorované sítě. Můžeme je klasifikovat podle místa, kde detekce probíhá.

5.2.1 Způsob klasifikace dle místa detekce

Network-based IPS (NIPS): založeno na monitorování síťového provozu, který je pak analyzován.

Wireless IPS (WIPS): monitoring probíhající na bezdrátových sítích. Ten je stejně jako ostatní analyzován a vyhodnocován.

Network behavior analysis (NBA): analýza chování síťového provozu. Provádí se analýza odmítnutého provozu na firewallech, neobvyklý nárůst přenosu dat mezi síťovými prvky.

Host-based IPS (HIPS): jde o monitorování jediného klientského počítače, kdy se na dané zařízení nainstaluje monitorovací aplikace a ta pak analyzuje veškerý podezřelý provoz právě na této stanici.

5.3 SNORT – aplikační řešení NIDS

Snort je aplikací, která má za cíl detekovat útoky v síti a odposlouchávat provoz na síti. Snaží se najít v procházejících paketech vzorky známých útoků. Je možné jej nastavit tak , aby ve chvíli, kdy je nalezena shoda, provedl různé akce. Provoz však nepřerušuje. Snort je možné spustit na velké škále operačních systémů. Spustitelný je v OS Linux, Windows NT/2000/XP, Unix (Solaris, *BSD, a jiných).

5.3.1 Režimy práce

Deamon Snort může běžet ve třech základních režimech, ty můžeme pak ještě upřesnit spoustou přepínačů zadaných přes terminál:

- Sniffer: přenáší na standardní výstup, (nejčastěji monitor) pakety, které *„tečou“* přes monitorovaná síťová rozhraní.
- Logger: ukládá pakety na disk do logových souborů. Jde o obdobu modu Sniffer s rozdílem, že procházející pakety jsou navíc ukládány.

- NIDS: na základě politiky definované v konfiguračním souboru rozhodne, co se s paketem udělá a jaká bude další reakce.

5.3.2 Moduly Snortu

Snort je řešen modulárním způsobem. Moduly fungují všechny dohromady. Výsledkem jejich práce je pak detekce útoku a vygenerování potřebné reakce na daný útok.

Rozdělit jej můžeme takto:

- Jednotka paketového záchytu
- Zásuvné moduly preprocesu
- Hlavní detekční jednotka
- Systém logování a výstrah
- Výstupní zásuvné moduly

5.3.3 Konfigurace

Jako u většiny Linuxových aplikací se všechny konfigurační soubory produktu vyskytují v adresáři /etc/. Snort je má umístěny ve složce /etc/snort a logy jsou uloženy v adresáři /var/log/snort.

Konfiguračních souborů SNORTU je hodně, ale v můžeme je lze rozdělit do 3 skupin:

1. hlavní konfigurační soubory
2. soubory s pravidly pro IDS
3. vlastní konfigurační soubory

Hlavní konfigurační soubor snort.conf obsahuje základní nastavení a je rozdělen na 4 části:

1. hlavní definice, které démona informují o topologii kontrolované sítě a o základních aplikačních serverech
2. podrobnější nastavení snortu – které moduly pro dekódování provozu použít, kolik jim přidělit paměťového prostoru a které protokoly sledovat.
3. prioritizace výstrah (alertů) a zdroje pro vyhodnocování chyb (defaultně v souborech classification.config a reference.config)
4. nastavení jaké pravidla (skupiny pravidel) se mají použít při detekci (implicitně v souborech *rules)

5.3.4 Pravidla

Pravidla můžeme uspořádat stejně přehledně, jako tomu je u pravidel implicitních definovaných už při instalaci. Pro pravidla definovaná přímo uživatelem systému je připraven prázdný soubor local.rules. Pro přehlednost se nepřipisují do pravidel vytvořených při instalaci, ale do vlastního souboru.

Kompletní popis pravidel by vyžadoval rozsáhlý prostor, proto si vysvětlíme jen jedno vybrané, které je najdeme v implicitním souboru pravidel icmp.rules (celý text příkazu je na jediném řádku):

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Large ICMP
Packet"; dsize: >800; reference:arachnids,246; classtype:bad-
unknown; sid:499; rev:3;)
```

Pravidlo detekuje paket icmp echo request s podezřele velkým datovým segmentem. Délka datového segmentu větší než 128 bytů je podezřelá a může znamenat například na DoS útok nebo komunikaci s již napadeným serverem.

Jednotlivé části zamenají:

- Klíčové slovo **alert** říká, že pravidlo vygeneruje alert zvolenou metodou a paket zaloguje.
- protokol **ICMP** – Nastavení na jakém protokolu má snort analýzu udělat (v současnosti je snort schopen analyzovat ICMP, UDP a TCP).
- Text až k levé kulaté závorce specifikuje zdroj a cíl (včetně portu, pokud to má smysl) paketu. V tomto případě musí ping přijít z externí sítě.
- V závorce je popsáno vlastní pravidlo, jednotlivé hodnoty jsou od sebe odděleny středníkem a jsou ve formátu <název_pole>:<hodnota>;
- **msg** – zpráva pro alert a log, název zprávy
- **dsize** – velikost datového segmentu paketu
- **reference** – kde lze nalézt informace o dané signatuře, více v konfiguračním souboru reference.config.
- **classtype** označuje klasifikaci události, v tomto případě neznámý, ovšem potenciálně velmi nebezpečný druh provozu.

Pravidla se mohou být mnohem komplexnějšími, lze jimi aktivovat jiná pravidla. (Například pokud sonda detekuje buffer overflow, může aktivovat pravidlo, které zaznamená dalších x paketů následující komunikace).

5.4 Systém Honeypot

Jedná se o systém, který má za úkol přitahovat potencionální útočníky a monitorovat jejich chování. Honeypot se využívá nejčastěji k detekci malwaru a jeho následnou analýzu. Malware velmi často mění strategii útoku a vyhýbá se detekci IPS systémy. Proto je potřebné malware přesvědčit o tom, že je v bezpečí při páchání své škodlivé činnosti a současně s tím jej detekovat. Detekce je pak důležitá pro aktualizace antivirů a jiných zabezpečovacích systémů. Detekce vždy sice přijde později,

než samotný malware, ale i tak je důležitá, protože stačí jedno místo, na kterém bude malware detekován a prostřednictvím sdílení znalostní báze na uzlových serverech pak může být zabráněno k průniku do jiných ještě nenapadených sítí.

Možná existence honeypotu už zneklidní hackera natolik, že si nikdy nemůže být jist, jestli jej právě nějaký nesleduje. Vždy totiž existuje možnost, že při průniku do systému budou chyceni a následný stejný útok už se většinou nepodaří, protože tuto aktivitu honeypot už vyhodnotí jako útok.

Honeypoty detekují činnost neoprávněných požadavků přicházejících na systém. Detekce je plně automatická a sbírá veškerá možná data o útočníkovi. Detekce na základě sesbíraných dat bud vyloučí, že jde o útok, nebo naopak to potvrdí. Takto sbíraná data jsou pak rychleji vyhodnocována, než kdyby byly sbírány data pouze jen z napadených systémů.

Honepot není většinou na síti jeden samotný systém, ale jedná se většinou o kompletní síť Honeypots. Jednotlivé Honeypoty mezi sebou sdílí informace o malwaru a provedených útocích.

Honeypoty můžeme dělit do skupin podle interakce, a to buď podle míry interakce, nebo směru interakce.

Dělení podle míry interakce:

- **s nízkou mírou interakce** – systémy, které simulují pouze pár funkcí transportní vrstvy operačního sytému. Neznamená to, že honeypot v něm musí být instalován. V takovýchto systémech je pak jednoduché identifikovat známé hrozby. Nelze však detekovat nové druhy útoků.
- **s vysokou mírou interakce** – systémy, které nabízejí útočníkovi kompletní reálný systém se všemi službami a funkcemi. Bohužel tento způsob interakce nabízí útočníkovi možnost napadnout celý systém včetně honeypotu. Je tedy jasné, že údržba takovéhoto systému je pak složitější. Velkou výhodou ale zůstává možnost detekovat i nové druhy útoků.

Dělení podle směru interakce:

- **Serverové honeypoty** - jde o nejčastější honeypoty. Nejčastěji jsou pasivními pozorovateli a vyčkávají na útočníka. Jsou schopny zpracovávat velké množství požadavků. Nejčastěji k detekci červů a exploitů síťových služeb.
- **Klientské honeypoty** – útoky hackerů jsou nejčastěji vedeny právě na klientské stanice, na kterých je možné najít velké množství cenných dat, proto vznikly i klientské honeypoty. Mají za cíl se chovat jako samotný uživatel a tak procházejí internetové stránky a detekují, zda nedošlo ke změně integrity. Mezi nejčastější hrozby patří phishing, nebo chyby v prohlížečích. Analýza pak probíhá pomocí jednoduchých metod honeypotu nebo jsou k ní využity programy třetích stran, jako jsou např. antiviry nebo sandboxy určené pro tuto analýzu.

5.5 HONEYD – softwarové řešení honeypotu

Honeyd je malý daemon, který vytváří virtuální zařízení v síti. Virtuální zařízení můžou být nakonfigurována se spuštěnou libovolnou službou a jejich reakce nastaveny tak, aby se jevily jako služby na jakýchkoli operačních systémech. Honeyd zlepšuje kybernetickou bezpečnost tím, že zajišťuje mechanismy pro rozpoznávání a posouzení ohrožení. Také odrazuje útočníky tím, že skryje reálné systémy ve středu virtuálních systémů, které díky svému nastavení mohou přitahovat větší pozornost.

Zařízení se jeví jako zcela reálné. Je možné pomocí příkazu ping nebo traceroute kontaktovat virtuální stroje. Každý typ služby na virtuálním počítači lze simulovat jednoduchým konfiguračním souborem.

Honeyd může být použit na vytvoření virtuální (fiktivní) sítě nebo může sloužit jen pro obecné monitorování sítě. Podporuje vytvoření virtuální síťové topologie, včetně specializovaných cest a routerů. Síťové cesty lze nastavit s latencí a ztrátou paketů, aby se topologie zdála realističtější.

5.5.1 Stručný přehled funkcí, které Honeyd podporuje:

- Simuluje tisíce virtuálních počítačů zároveň.
- Je možná konfigurace libovolných služeb pomocí jednoduchého konfiguračního souboru:
    - Umožňuje proxy připojení.
    - Pasivní snímání otisků prstů k identifikaci vzdálených počítačů.
- Simuluje operační systémy na protokolu TCP / IP:
    - Mate utility nmap a xprobe,
    - Možnost nastavit FIN-scan politiku.
- Simulace libovolného směrování topologií:
    - Má konfigurovatelné latence a packet loss.
    - Je schopen asymetrického routingu.
    - Je schopen integrace fyzických strojů do topologie.
    - Umožňuje distribuované Honeyd přes GRE tunely.
- Subsystém virtualizace:
    - Můžou na něm běžet virtuální služby tvářící se jako reálné, ty pak odpovídají jako klasické UNIXové služby. Např.: Web servery, FTP servery, atd. ..
    - Dynamické vázání portů do virtuálního adresového prostoru, atd.

5.5.2 Subsystém virtualizace

Jak je výše zmíněno Honeyd podporuje virtualizaci služeb provedením unixové aplikace jako subsystémy běží ve virtuálním prostoru s IP adresu konfigurovanou honeypotem. To umožňuje jakékoliv síťové aplikaci dynamicky svázat porty, vytvořit TCP a UDP spojení na virtuální IP adresu.

Subsystémy jsou virtualizace, které zachytí své síťové požadavky, a přesměrují je na Honeyd. Každá konfigurační šablona může obsahovat subsystémy, které jsou spuštěny jako oddělené procesy, pokud je šablona vázán na virtuální adresu IP. Další výhodou tohoto přístupu je schopnost vytvářet honeypots sporadický provoz na pozadí který vypadá jako vyžádání webové stránky, čtení e-mail, atd.

5.5.3 **Zabalená síť**

Honeyd podporuje asymetrické trasy a její začlenění do fyzického stroje na virtuální síťové topologii. V důsledku toho je možné použít Honeyd pro jednoduché síťové simulace: Hostitelé mohou být vystaveny vysoké latenci linek nebo ztrátě paketů, ruznému směrování infrastruktury apod.

Konfigurace jednoduché struktury může vypadat takto:

```
route entry 10.0.0.1 network 10.0.0.0/8
route 10.0.0.1 link 10.0.0.0/24
route 10.0.0.1 add net 10.4.0.0/14 tunnel "thishost" "honeyd-b"
route 10.0.0.1 add net 10.1.0.0/16 10.1.0.1 latency 55ms loss 0.1
route 10.0.0.1 add net 10.2.0.0/16 10.2.0.1 latency 20ms loss 0.1
route 10.0.0.1 add net 10.3.0.0/16 10.2.0.1 latency 20ms loss 0.1
route 10.1.0.1 link 10.1.0.0/24
route 10.2.0.1 link 10.2.0.0/24
[...]
route 10.2.0.1 add net 10.3.0.0/16 10.3.0.1 latency 10ms loss 0.1
route 10.3.0.1 link 10.3.0.0/24
route 10.3.0.1 add net 10.3.1.1/24 10.3.1.1 latency 10ms
route 10.3.0.1 add net 10.3.240.0/20 10.3.240.1 latency 5ms
route 10.3.1.1 link 10.3.1.1/24
route 10.3.240.1 link 10.3.240.0/20
route 10.3.240.1 add net 0.0.0.0/0 10.3.0.1 latency 40ms loss 0.5
[...]
bind 10.2.0.243 to fxp0
bind 10.3.1.15 to fxp0
```

5.6 Deamon Syslog-ng – vhodný logovací nástroj na síti

Syslog-ng deamon (syslog-ng = syslog new generation) služba, která je schopná sbírat a zpracovat všechny logované události na pracovních stanicích a serverech. Její velkou výhodou je, že může běžet i jako logovací serverová aplikace Samotná aplikace však logy neanalyzuje. Škatulkuje, třídí a upravuje je tak, aby byl dostatečně přehledné a rychle se v logovaných zprávách orientovalo. Stejně jako výše popisovaný produkt SNORT má taky podporu ve velkém množství OS (GNU/Linux, *BSD, Solarix, AIX, HP-UX, MS Windows, …) atd.

Syslog-ng je dostupný ve třech provedeních: Open Source Edition, Premium Edition a Store Box Edition. Druhé dvě varianty (Premium a Store Box) mají proti open source variantě pár rozšíření. (podpora SSL/TLS, ukládaní logů do databází, atd.).

Pokud budeme logovat události pomocí syslovu-ng na desktopových stanicích nemusíme ve výchozím nastavení aplikace nic měnit. Pokud budeme ale chtít logovat události na serverech je situace jiná. Na serverech pracujeme s velkým množstvím logovaných informací z mnoha aplikací důležitých pro chod serveru.

Servery jsou velmi citlivé na spolehlivost. Důležitá je maximální dostupnost a bezpečnost serverových aplikací. Ve chvíli, kdy se vyskytne problém, tak právě logové informace bývají nejčastějším zdrojem detekování problému. Je proto důležité dobře vybrat logovací server. Myslím si, Syslog-ng je pro tento účel velmi dobrou volbou.

Jak již bylo zmíněno, syslog-ng neanalyzuje logy a tak je metoda prohlížení a analýza zpráv na volbě správce systému. Tento fakt se dá eliminovat použitím některé aplikace (například logwatch) pro analýzu logů, která periodicky prohlíží logy podle konfigurace a reportuje, například zasíláním emailů, případné zjištěné problémy, které jsou obsažené ve zprávách. Tímto způsobem se tak dá alespoň částečně zautomatizovat proces zpracování a analýzy logů. Částečně proto, že některé aplikace si spravují logy samy a zaspisují zprávy do různých souborů a v různých formátech. V takovém případě je potřeba např. přidat do syslog-u novej zdroj, který odpovídá logům pro danou aplikaci nebo na straně aplikace upravit posílaní zpráv do souboru/socketu, který je zdrojem zpráv syslogu.

5.6.1 **Příklad nastavení**

Příklad nastavení uvádím pro představu, jak jednoduše se dá spustit logování po síti.

Server pro sběr logů můžete nastavit celkem jednoduše, je třeba udělat pár změn v souboru /etc/syslog-ng/syslog-ng.conf

- Definice zdroje dat ze sítě:

```
source s_net {
        udp();
};
```

Použijeme UDP z důvodu zpětné kompatibility se starým syslogem. Lze udělat více zdrojů, třeba pro každý server, který bude na loghosta logovat jeden zdroj. Pak se to zadává pomocí příkazu:

```
udp(ip(10.0.0.1) port(514));
```

Nadefinujeme samostatný soubor, do kterého se budou ukládat logy ze sítě.

```
destination d_net { file("/var/log/net.log"); };
```

Nakonec nastavíme vlastní logování dat ze sítě do samostatného souboru

```
log {
        source(s_net);
        destination(d_net);
```

```
};
```

Cílový soubor pro logy můžete udělat pro každý stroj samostaný:

```
source s_net {udp();};
destination d_net{ file("/logy/$HOST/$YEAR/$MONTH/$FACILITY-
$YEAR$MONTH" \
owner(root) group(root) perm(0600) dir_perm(0700)
create_dirs(yes));};
log {
        source(s_net);
        destination(d_net);
};
```

Lze nastavit i logování po síti protokolem TCP

```
tcp(ip(0.0.0.0) port(5000));
```

Nastavení na straně klienta také není nejtěžší. Buď nadefinujete jako cíl UDP

```
udp("loghost" port(514));
```

nebo TCP

```
tcp("loghost" port (5000));
```

To by mělo pro chod stačit.

5.7 Aplikační řešení Zabbix

Dohledový systém ZABBIX je systém pro sběr dat založený na open source. Jeho hlavní výhodou je jeho variabilita. Pomocí šablon lze sledovat jakékoli zařízení připojené na síť, popřípadě jakoukoli službu běžící na zařízení, které je dostupné prostřednictvím počítačové sítě. Celý systém je možné stáhnout na stránkách http://www.zabbix.org, kde je k dispozici i dokumentace celého systému. Je v ní popsáno, jak lze systém nainstalovat do operačního systému. Stejně tak jsou zde popsány první kroky se systémem.

5.7.1 **Tři způsoby monitorování pomocí dohledového systému Zabbix:**

5.7.1.1 ***Monitoring prostřednictvím agenta instalovaného na klientské PC***

Klientský software monitoruje PC, prostřednictvím aktivní kontroly, kdy v pravidelných intervalech je na PC spuštěna kontrola, která provede sérii testů. Ty následně nahraje na zabbixový server. Dle vyhodnocení výsledků pak zabbixový server hlásí chyby a posílá upozornění.

5.7.1.2 *Monitoring prostřednictvím aktivních kontrol zabbixového serveru*

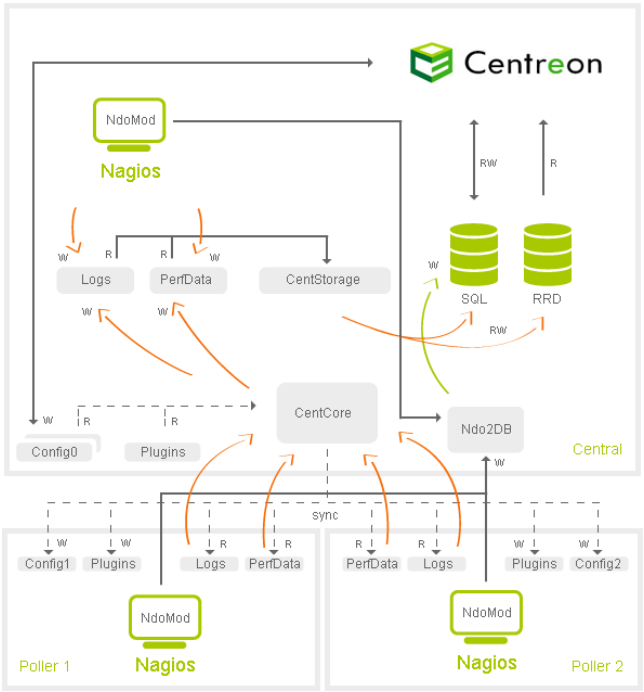
Na serveru zabbix jsou v pravidelných intervalech spouštěny úlohy kontroly aktivních služeb monitorovaných zařízení. Na serveru je možné sledovat dostupnost služeb, jako jsou ssh, smtp, ping (odezva), http, ftp a další. Tento monitoring má výhodu v tom, že není třeba na daném zařízení nic instalovat. Nevýhodou však je, že kontrolujeme dostupnost služby na dálku, a to z jednoho místa. Na výsledek kontroly tedy může mít vliv více aspektů, než jen samotná kontrolovaná služba. Výsledek můžou ovlivnit firewally a výpadky konektivity. Přesto je tato kontrola mnohdy dostačující, aby nám řekla vše, co potřebujeme vědět.

5.7.1.3 *Monitoring prostřednictvím SNMP protokolu*

Opět se jedná o vzdálený přístup k monitorovanému zařízení. Podmínkou tohoto monitoringu je, aby monitorované zařízení podporovalo komunikaci prostřednictvím protokolu SNMP. Pak se prostřednictvím tohoto protokolu dá monitorované zařízení nejen kontrolovat, ale i nastavovat. V případě nějaké události je možné na ni reagovat, což je velkou výhodou tohoto monitoringu.

5.8 Aplikační řešení Centreon

Systém Centreon je, stejně jako systém ZABBIX, software s otevřeným zdrojovým kódem typu open source. Za jeho hlavní klad se dá považovat grafická přehlednost a uhlazenost. Celý systém



Obrázek 1 - Schéma systémuCcentos

pracuje nad platformou Linux, takže na sítích s OS Windows je třeba do sítě zařadit alespoň jeden server s operačním systémem Linux nebo je možné jej instalovat na virtuální stroj.

Můžeme říct, že Centreon je pouze nadstavbou nad systémem Nagios, ale to je jen přiblížení. Systém Centreon dnes už Nagios tak integroval do své struktury, že je více relevantnější tvrzení, že Nagios je součástí Centreonu. Systém Nagios, stejně tak jako Centreon, i ZABBIX monitoruje síťové prvky s tím rozdílem, že pro Nagios nebylo nikdy vyvinuto samostatné grafické konfigurační a administrační rozhraní, takže jeho konfigurace byla vždy časově náročnější a vyžadovala odborné techniky, kteří byli schopni pracovat s konzolí operačního systému.

Centreon je modulární systém, který dnes disponuje moduly:

- Centreon Engine - nabízí alternativu k centrálnímu Nagios systému, který je hnacím motorem. Je vytvořený z Nagios verze 3.2.3. Cílem bylo zlepšit výkonnost Nagiosu a přizpůsobit jej tak, aby kompletně zapadl do Centreonu. To se taky stalo.
- Centreon Broker - nabízí efektivní způsob, jak uložit Nagios události v databázi. Díky pružnému jádru, může správce nyní rozhodnout, kudy přesně poteče datový tok přes síť. Tyto převody jsou díky tomu pak extrémně rychlé. Je navíc možné si nastavit pomocnou databázi, která bude sledovat pouze část IT infrastruktury. Taky je možné nastavit záložní databázi, která se aktivuje, když primární Nagios databáze je vypnuta. Tím se zajistí provoz Centreonu bez ztráty uživatelských dat.
- Centreon Network Monitoring - modul pro monitorování sítě. Může být použit k měření dostupnosti a výkonu switchů, firewallů, zatížení balancerů a routerů a aktivních prvků. Podporuje analýzu provozu včetně měření provozu v úzkých místech.
- CLAPI - Command Line API umožní správci ovládat Centreon z příkazového řádku. Modul dává správci možnost přidávání, úpravu, ale také odstranění všech konfigurací objektů Centreonu (hostitelé, služby, kontakty, skupiny). Konfiguraci z příkazové řádky je možné dělat dvouúrovňově a to tak, že nejprve se nadefinují potřebné parametry a pak je teprve vygenerována konfigurace pro Nagios. Pak už se jen provede restart Nagiosu přímo pomocí CLAPI. Modulem lze spravovat více Nagios subsystémů.
- Syslog - modul umožňuje zobrazovat události sítě z protokolu Syslog v jednom rozhraní. Opírá se o databázi, která je plněna službami, jako je syslog-ng nebo rsyslog.

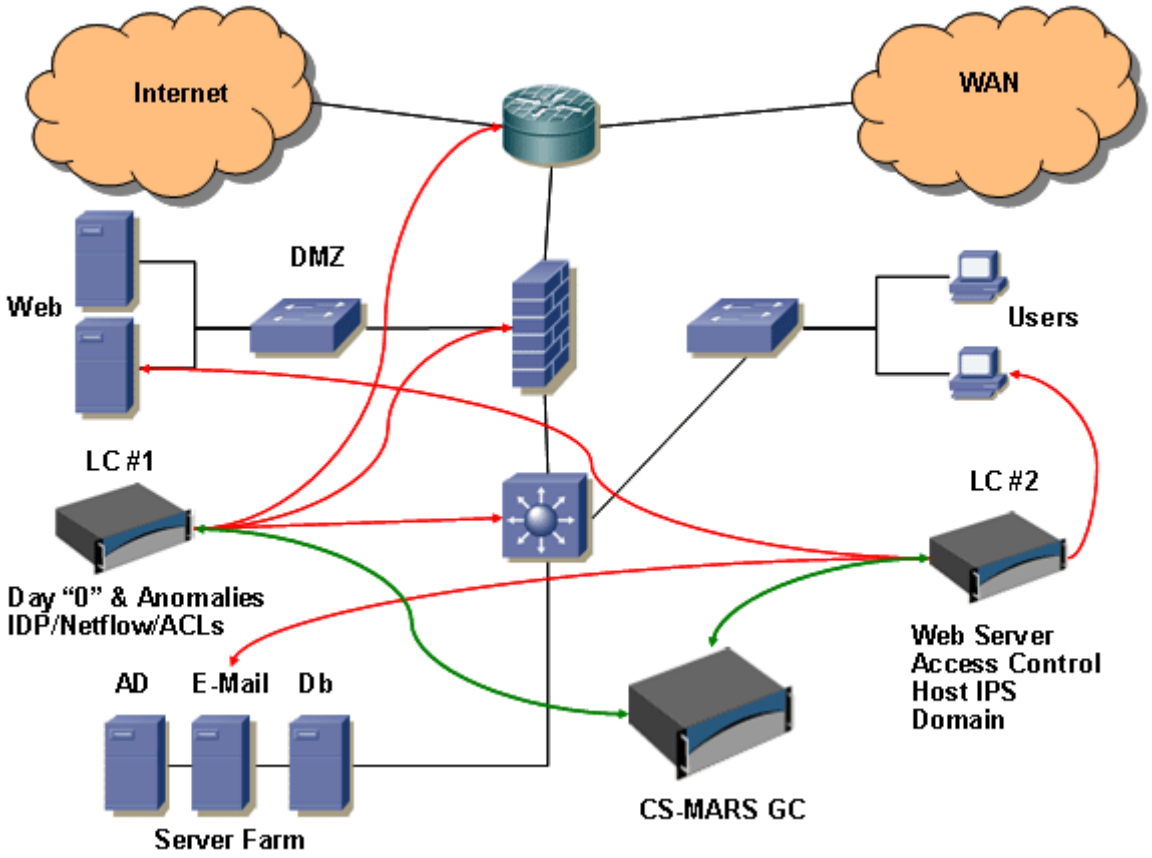
Centreon je velmi robustním řešením pro větší firmy, které udržují v chodu desítky síťových zařízení. Jeho modularita má velmi pozitivní vliv na sestavení systému přesně na míru potřebám.

5.9 CISCO řešení MARS

Jedná se o velmi robustní řešení postavené na hardware. Bohužel v současné době již společnost CISCO řešení MARS (Monitoring, Analysis and Response Systém) nepodporuje. Jeho vývoj byl zastaven.

Přesto jsem si inspiraci z tohoto řešení dovolil vzít taky. Podle schématu na obrázku je vidět, že v síti jsou nasazeny tři hlavní prvky, které mezi sebou vzájemně komunikují (zelené šipky). Dva ze tří zmiňovaných prvků (LC #1 a LC #2) zaznamenávají provoz na veškerých síťových zařízeních a následně jej dávají k dispozici MARS motoru, který vše vyhodnocuje a na dané problémy reaguje. Reaguje opět tak, že prostřednictvím prvků LC #1 a LC #2 nastavuje restriktivní opatření na uživatelských počítačích, routerech, firewallech atd.

Řešení je pro nasazení jednodušší než aplikace Zabbix nebo Centreon. Jednoduchost nasazení řešení MARS ale zákazník pocítí na peněžence. Je to dáno tím, že jde o komerční a ne open source řešení, jak je tomu u Zabbixu nebo Centreonu.



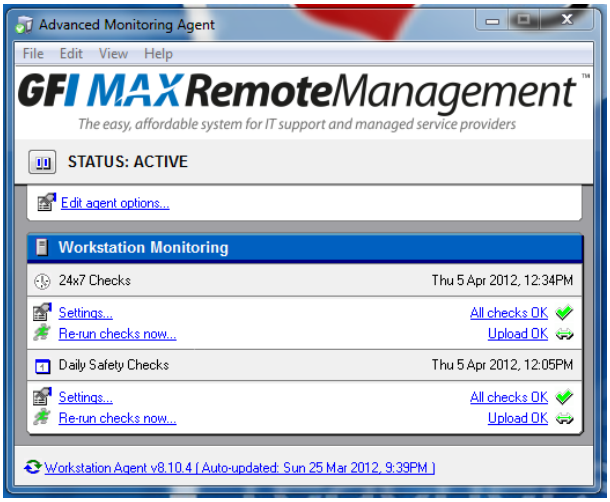
Obrázek 2 – Schéma nasazení systému MARS od společnosti cisco v síti

5.10 Aplikační řešení GFI Remote Management

Řešení společnosti GFI. Pro monitoring počítačových stanic na síti vyvinula společnost GFI systém nazvaný GFI Remote Management, který pomocí agentů instalovaných na jednotlivých počítačích monitoruje události jednotlivých stanic. Proti předchozím systémům není schopen monitorovat jiné zařízení než počítačové stanice. Svůj handicap však velmi dobře nahrazuje možnostmi, které nabízí pro vzdálenou správu PC. Nejedná se tedy jen o monitorovací systém, ale i o systém, kterým je možné spravovat velké množství stanic jednoduchými příkazy.

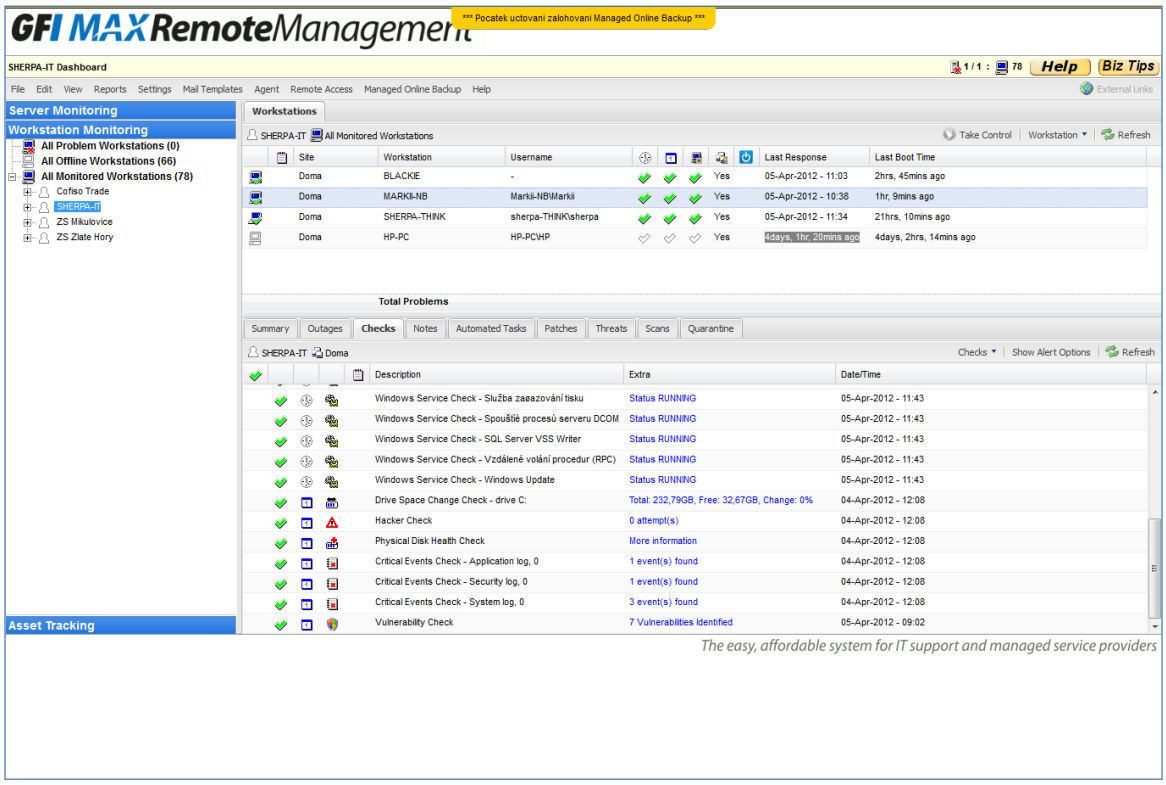
Celý monitoring je možné sledovat přes centrální panel dostupný prostřednictvím webového prohlížeče, z kterého pak je možné sledované zařízení na dálku spravovat, ať už přes vzdálenou plochu, či pomocí příkazů pro aktualizaci nebo automatických skriptů.

Agent instalovaný na jednotlivých stanicích kontroluje systém podle předem administrátorem nastavených kontrol. Kontroly jsou rozděleny do jednotlivých skupin podle důležitosti a typu. Ty, které nejsou tak důležité, se kontrolují v pravidelném intervalu buď dvanácti nebo dvaceti čtyř hodin. Kontroly, které jsou důležité a kontrolují části systému, na kterých závisí chod, jsou monitorovány častěji a to buď v intervalu jedné hodiny nebo třiceti minut.



Obrázek 3 - Agent GFI instalovaný na PC

Každý agent kromě pravidelných kontrol umí sledovat i hardwarové vybavení PC, takže administrátor pak má možnost zjistit, který počítač na síti má nejslabší konfiguraci a tak efektivně řešit upgrade hardwaru v monitorovaných sítích.



Obrázek 4 - Rozložení centrálního panelu

Agent umí komunikovat i opačným směrem, a to od centrálního serveru směrem k počítači. Z centrálního pultu se dají vyvolat příkazy pro restart stanice a spustit jakýkoli předem definovaný skript. Díky této možnosti je možné na PC provádět jakékoli úpravy, a to i hromadně na všech stanicích současně.

Centrální pult (dashboard) je rozdělen do tří částí:

1. část, ve které se zobrazují seznamy klientů a jejich sítí
2. část, ve které nalezneme všechna zařízení jednotlivé sítě
3. část, ve které se zobrazují informace o jednotlivých stanicích

viz: obrázek 3

Administrátor prostřednictvím centrálního pultu má ještě spoustu dalších možností, jak nastavit chování počítačů. Od nastavení pravidelných kontrol počítače antivirem, přes správu aktualizací systému a instalovaných programů, až po vzdálený přístup pomocí integrovaného programu Team Viewer. Tuto funkcionalitu ve své aplikaci v současné chvíli implementovat nechci.

6 Zjištění z prostudovaných systémů

Ze všech prostudovaných systémů vyplynula řada závěrů. Bylo zjištěno velké množství pozitivních vlastností a sofistikovaně zpracovaných problémů, které systémy řeší, ale, jak bylo možné očekávat, bylo nalezeno rovněž mnoho negativních stránek. Před zahájením mého projektu, bylo třeba ujasnit, co je na výše popsaných systémech dobře řešeno, a co by bylo možné využít pro moje potřeby, a co naopak příliš dobrým řešením není a raději se toho vyvaruji.

6.1 Jak využít SNMP?

Monitorování prostřednictvím protokolu SMTP je sice dobře propracováno, ale v prostředí, kde chceme vyhodnocovat útoky na síť, či zda nedochází k výpadkům nějaké služby, se tento protokol nejeví jako nejvhodnější. Jeho podpora na síťových prvcích je vysoká. Najdeme ji jak u switchů, routerů, tak u tiskáren i počítačů. V kapitole o SNMP bylo zmíněno, že jde o protokol, který je možné jak monitorovat, tak i nastavovat. Většinou pro monitoring zařízení můžeme zvolit jinou (lepší) alternativní metodu a pro nastavování jednotlivých zařízení výrobce vždy nabízí i své řešení, které je jednodušší, než SNMP. Z těchto důvodů jsem si místo pro získávání informací prostřednictvím protokolu SNMP ve svém řešení ponechal (viz dále).

6.2 NetConf má dnes šanci?

Dle mého názoru je šance NetConf na další masivní využití diskutabilní. Jedná se o velmi zajímavý protokol. Má však jednu potíž. Nenachází dostatečnou podporu u výrobců. Ti většinou buď implementují své řešení, nebo využívají zavedeného a roky ověřeného standardu SNMP, který najdeme prakticky opravdu skoro ve všech zařízeních, která je možné nějak spravovat. Aby NetConf měl dostatečnou šanci, muselo by jej začít využívat větší množství správců. Proto jsem se jeho využití rovněž vzdal a ve svém projektu toto řešení nepoužívám. Chci vytvořit aplikaci, která má za cíl monitorování a vyhodnocování provozu sítě a nikoli vzdálenou konfiguraci.

6.3 NetFlow je vhodné pro ISP (poskytovatele připojení k internetu)

Vhodnost protokolu NetFlow od společnosti CISCO pro monitoring sítě je omezen pouze na prvky od této firmy. Vzhledem k tomu, že CISCO řešení jsou sice kvalitní, ale i drahá, v prostředí menších firem, škol a neziskových organizací se s nimi moc nepotkáme. Proto využití protokolu NetFlow nemůžeme brát z pohledu návrhu systému, který bude provozovatelný na co nejvíce uživatelských sítích, jako nevhodný. Jeho alternativou je protokol TraficFlow od společnosti Mikrotik, která nabízí na trhu levnější varianty routerů vhodných do menších organizací. Tento protokol je natolik totožný s NetFlow, že je možné využít pro sběr informací, jak z prvků CISCO, tak z prvků

Mikrotik stejný kolektor. Nevýhodou je slabá softwarová podpora, která by nabízela rozumné řešení pro zobrazování získaných informací.

6.4 Zabbix na linuxových sítích králem

Zabbix je, jako open source řešení, řešením hodně zajímavým. Aplikace, která má velkou komunitní základnu, a proto i jeho uvedení do provozu nepatří k nejsložitějším. Na svých webových stránkách navíc nabízí i hotové instalace do virtuálních serverů od VMware nebo Oracle (Virtual Box). Problematická je u něj konfigurace, při níž systému neznalý člověk začne tápat nad nastavením monitorovacích šablon, kterých je v systému velká řada, jejich názvy jsou uvedeny ve zkratkách a informací o druhu a způsobu nastavení šablon je mnoho. Stejně tak se mi na systému nelíbil způsob instalace a konfigurace zabbixového agenta v prostředí operačního systému Windows. Za výhodu tohoto systému pokládám, že je napsán v kódu PHP, který je jednoduše spustitelný na jakémkoli webovém serveru. Definitivní rozhodnutí, proč tento systém nenasazovat na síti, ovlivnil fakt, že jeho konfigurace byla neúměrně složitá jeho užitku a navíc zpravidla nenajdete síť, která by měla většinu klientských stanic s operačním systémem Linux.

6.5 GFI Remote Managment a jeho přínos

Firma GFI mě svým řešením velmi oslovila. Na toto řešení jsem narazil, když jsem hledal pro mnou spravovaný server na základní škole kvalitní spamový filtr. Lidmi, z jedné nejmenované firmy, kteří nám dodávali wifi přístupové body, mi byl doporučen produkt od firmy GFI s názvem GFI MailEssentials. Následně jsem pak po zjištění, že jde o kvalitní produkt, začal sledovat i jiné produkty této firmy a pro mě vhodný přínos měla právě výše jmenovaná aplikace. GFI Remote Managment mi vyřešil problém dohlížení a monitorování velkého množství stanic a počítačů nejen u zákazníků, ale i u známých, kteří po mě vyžadovali běžnou údržbu PC, která zabere značné množství času.

Remote Managment je uceleným řešením, které se neustále rozvíjí, za velmi příjemnou cenu. Cenová politika firmy GFI, kdy platíte jen za služby (monitoring), který využíváte je velmi rozumná. Dohlížení takové stanice vychází v současné době na maximálně 20 korun za měsíc. Za tuto cenu je zajištěna pravidelná aktualizace stanice. Kontroly všech možných typů, včetně dostupnosti a vzdáleného dohledu i nad antivirem. Jde o řešení, které nejenom kontroluje stav počítače, ale i na dálku je schopno velmi dobře nastavovat a jednoduše spravovat.

Jeho velkou nevýhodou je, že lze monitorovat pouze počítačové stanice a servery, navíc jen ty, na kterých běží operační systém Windows.

Myslím si, že v kombinaci s jiným produktem, který je schopen monitorovat ostatní síťové prvky, jde o velmi silné řešení.

6.6 Honeyd deamon

Velkou výhodou tohoto démona je jeho jednoduchost konfigurace a začlenění do reálné sítě. Hodně mě u něj zaujala možnost šablon, kdy pro každou běžící službu na vizualizované síti a následně i stroji je možné použít šablonu. Díky ní se pak virtuální PC jeví jako zcela reálné a tím může mást potenciálního útočníka.

Jednoduchostí konfigurace, která se dá neustále rozšiřovat a vylepšovat stejně tak jako reálná síť nahrává možnosti zdokonalovat bezpečnostní řešení. Proto při sestavování vlastního řešení jsem se k využití démona Honeyd uchýlil taktéž. Viz další strany mé diplomové práce.

6.7 Syslog-ng

Nejprve jsem hledal aplikační řešení logového serveru, které by bylo spustitelné pod systémem Windows, protože na sítích, které spravuji, běží převážně Windowsové servery. Našel jsem řešení v podobě programu Kiwi Syslog server, které mi nakonec bylo rozmluveno garantem společně se spoustou jiných nápadů, které jsem měl. Při myšlence virtualizovat celé zkušební řešení mě garant doporučil open-source aplikaci Syslog-ng. Po zběžném prostudování vlastností tohoto serveru jsem rád, že mi bylo toto řešení doporučeno. Jeho variabilita je velmi velká a dovoluje jednoduchým způsobem vytvořit přehlednou strukturu logovaných informací.

Syslog-ng v kombinaci s aplikací kterou jsem si vymyslel (viz níže). Se mi jeví jako velmi silný nástroj pro zaznamenávání událostí vytvářených síťovým prostředím.

7 Návrh vlastního řešení systému - Controleye

U jednotlivých systémů jsem vypsal, co se mi na nich líbilo a co jsem jako přínos neviděl. Na základě těchto zjištění jsem se rozhodl implementovat systém co nejvíc modulárně, tak aby bylo možné v případě změny hardware nebo struktury systému co nejjednodušším způsobem, popřípadě bez zásahu obnovit monitorování vyměněných prvků.

Celý monitoring by se dal rozdělit podle typu zařízení, které je třeba monitorovat do dvou velkých skupin. A to skupiny koncových zařízení a skupiny zařízení umožňujících síťový provoz. Tak situaci řeší i výše zmiňované systémy v první části této zprávy. Mnohé z nich se, bohužel, věnují jen jedné části, nad kterou provádí monitoring, takže kontrola stavu sítě není vždy kompletní. V lepším případě monitorují obě dvě části systému, ale jednu z nich jen velmi slabě, takže spíš nepřináší dostatečný užitek. Ve svém řešení jsem se snažil o monitoring obou skupin.

Monitoring obou skupin je velmi rozdílný. Na koncovém zařízení uživatel zpravidla potřebuje provozovat velké množství aplikací a služeb, které je třeba sledovat. Zatímco na zařízeních poskytujících síťovou konektivitu konkrétnímu zařízení potřebujeme monitorovat hlavně tok dat a bezpečnost provozu.

Tím jsou vymezeny dvě velké oblasti monitoringu, které bylo mým cílem sjednotit tak, aby v konečné fázi přinášely správci maximum možných informací v co nejmenším prostoru. A výrazem prostor je v tuto chvíli myšlen jak prostor časový, tak prostor fyzický.

Dalším důležitým bodem, o kterém jsem musel hodně přemýšlet, bylo jakou technologii zvolit, aby monitorovací systém bylo opravdu možno instalovat a provozovat na maximálním množství zařízení. Dospěl jsem k názoru, že se bude třeba opřít o právě jedno z monitorovaných médií a využít síťového prostředí.

V neposlední řadě bylo třeba se vyrovnat s velkým množstvím informací, které se v reálném čase sbírají ze všech síťových prvků. Musel jsem počítat s tím, že na síťových stanicích je možné během dne vytvořit až několik tisíc událostí. Stejně tak i router nebo switch, může za den vytvořit tisíce událostí.

Své řešení jsem se snažil co nejvíc navrhnout tak aby bylo lehce nasaditelné v jakékoli firemní síti. Za běžnou firemní síť považuji síť o 30 – 50 počítačových stanicích, několika řádově do 10 aktivních sítových prvcích a jednom až dvou serverech plnící role doménových kontrolerů, ftp,www, DNS, VoIP a souborových serverů (démonů).

Proto jsem se systém rozhodl navrhnout na následujících principech:

- modularita
- rozšířitelnost
- široká dostupnost

- nezávislost na hardware
- data senderu - informační vysílací systém datových paketů bez zpětné interakce.

Možností jak výše stanovených požadavků dosáhnout je několik. Ve svém řešení jsem zvolil následující cestu:

Aby bylo možné celý systém vzdáleně monitorovat, a nejlépe více sítí najednou, není v současnosti vhodnější cesty, než vytvořit centrální internetovou aplikaci, do níž by se sbíhaly informace ze všech monitorovaných zařízení. Internetová aplikace bude pracovat na webovém serveru, který bude dostupný v rámci sítě a pokud bude třeba i z internetu. Přístup k této aplikaci musí být zabezpečen tak, aby se k poskytovaným službám a informacím dostali pouze uživatelé s patřičným oprávněním přístupu, aby nedošlo ke zneužití informací.

Není výhodné, aby tato aplikace musela před finální informací pro správce ještě provádět transformace datových formátů. Internetová aplikace musí čerpat a zpracovávat standardně a pevně strukturovaná data splňující požadavky normálních forem, proto jsem se rozhodl vytvořit databázi, do které se budou v pravidelných intervalech načítat data prostřednictvím kolektorů. Databáze řeší jeden ze zmiňovaných problémů. A to způsob vypořádání s velkým množstvím informací, v nichž je nutno následně vyhledat informace důležité pro správu a efektivní řízení počítačové sítě a jejich prvků.

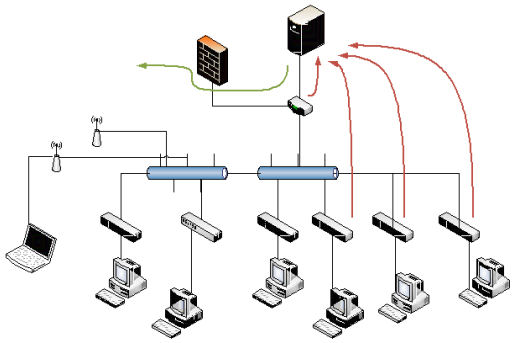
Všechen logovaný provoz se bude sbíhat na jedné serverové stanici, kterou můžeme nazvat logový server.

Na klientských stanicích s operačním systémem Windows jsem se rozhodl pro vlastní řešení. Pro tyto stanice je žádoucí napsat aplikaci, která bude mít za cíl posbírat maximální množství informací zaznamenaných v průběhu provozu stanic, a ty pak v pravidelných intervalech posílat ze stanice do databáze logovacího serveru. Protože bude třeba vyřešit, jak data nahrát na logovací server i v případě že bude mimo firemní síť, rozhodl jsem se využít pro odesílání dat strukturu XML souborů a následné nahrávání dat na server do centrální složky přes protokol FTP.

Na síťových prvcích jsem se rozhodl, že provozní informace budu sbírat prostřednictvím již nějakého hotového řešení. A to z důvodu, že tato řešení jsou dostupná v dostatečném množství v rámci různých serverových aplikací, které tento problém, řeší. Jedinou podmínkou, kterou je třeba dodržet je možnost kumulativně sbíraná data ukládat do centrální jednotné databáze.

Vzhledem k tomu, že požaduji systém, který bude možno instalovat u většiny uživatelů, bylo třeba vyřešit, jak přenést informace v případě že klient nemá vlastní server (centrální PC) ze všech síťových prvků mimo sledovanou síť tak, aby nebyla zbytečně moc vytížena linka, kterou je uživatel připojen do sítě internet. Řešení jsem našel v možném dvouúrovňovém sbírání informací, při němž v první úrovni sběru dat na uživatelské straně je možno nastavit, které informace se budou sbírat, a které ne.

Navržený systém pro sběr informací na switchích, routerech a stanicích linux využívající protokol syslog lze znázornit takto:

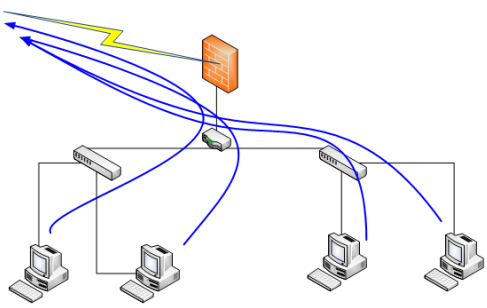


Obrázek 5 - Systém logování na server mimo firemní síť

Červené šipky znázorňují přenos logů ze switchů, routerů a linuxových stanic na *„zvolený firemní počítač“*. Ten musí být přístupný z internetu aby si lgovací server mohl stahovat data do centrální databáze, kde jej pomocí parseru centrální systém rozebere na části, které budou dále analyzovány a zpracovány.

Tento dvouúrovňový systém není zapotřebí u firem, které disponují vlastním serverovým hardwarem. V takovémto případě stačí, když *„zvoleným firemním počítačem“* bude server, na který bude možné nainstalovat všechny serverové aplikace nutné pro chod logovacího serveru.

V případě logovacího serveru mimo firemní síť budou z jednotlivých stanic s OS Windows události odesílány přímo na centrální logovací server bez mezistanice na *zvoleném firemním počítači“*, jak je to rozkresleno na obrázku níže. Směr datové zprávy ukazují modré šipky. Důvodem, proč se zprávy nesbírají stejně, jak z prvků na nichž je možné logovat prostřednictvím protokolu syslog (routery, switche, linuxové PC) je, že z každého počítače požaduji získání jiné informace, protože každý počítač slouží k jinému účelu. Dalším důvodem je vlastní logovací systém společnosti Microsoft (konfigurovat syslog do systému Windows by činilo systém sběru zbytečně složitý). Navržené řešení předpokládá poněkud větší náročnost práce při počátečním konfigurování stanic, která však bude kompenzována štíhlým přenosem přesných a užitečných informací, jež nebudou zahlcovat centrální databázi.



Obrázek 6 - Logové zprávy z windowsových stanic při dvouúrovňovém logování

V případě, kdy nebude třeba dvouúrovňové logování, se pouze na jednotlivých stanicích nastaví adresa firemního logového serveru.

Tímto řešením jsme dosáhli možnosti nasadit systém sběru událostí jak v sítích, které mají, tak v sítích, které nemají vlastní serverové stanice.

8 Realizace systému

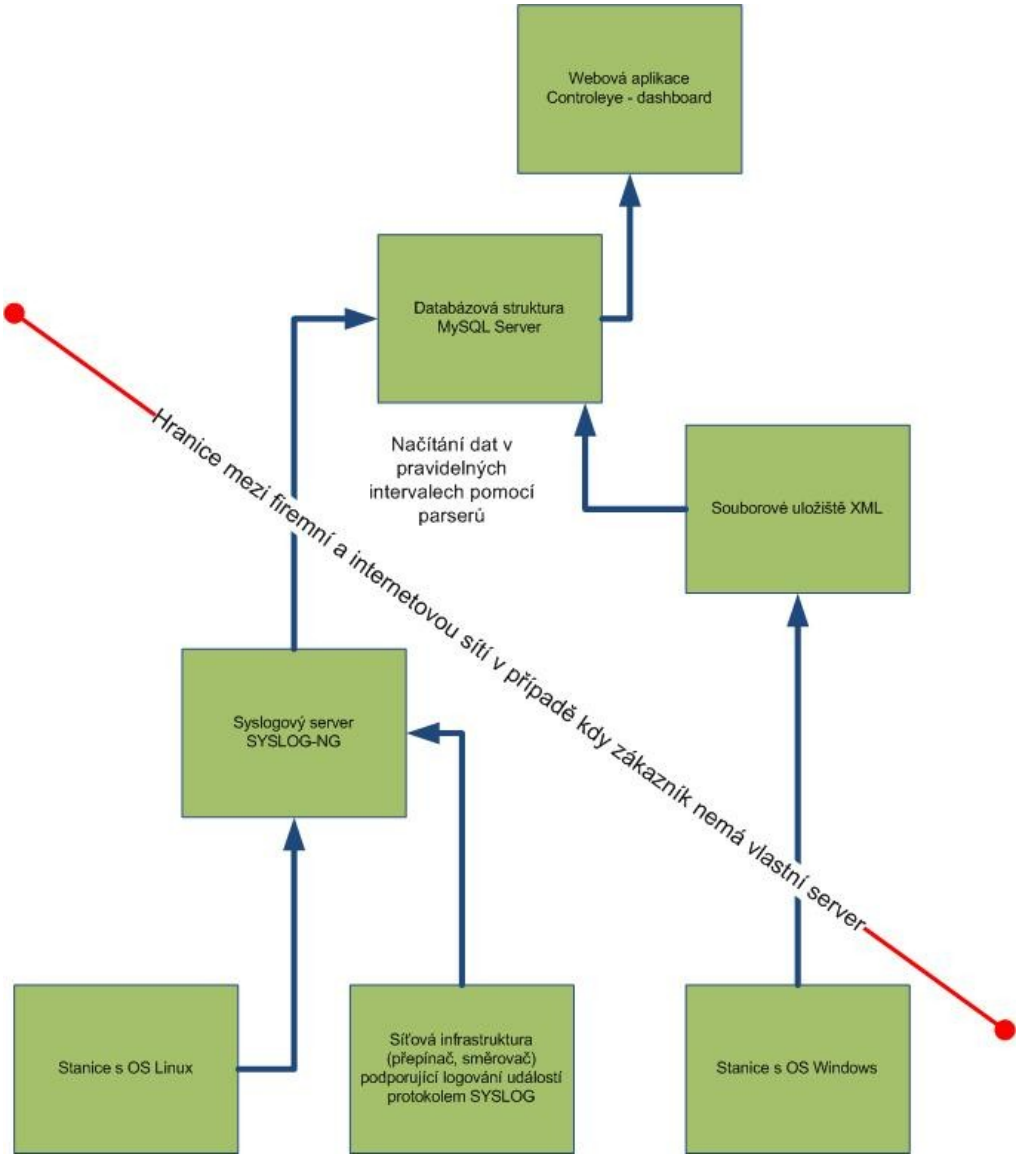
Vlastní řešení bylo implementováno v jazycích C# v prostředí .NET a PHP. V prostředí C# .NET byl implementován agent pro sběr informací na klientském PC se systémem Windows. V programovacím jazyku PHP byla implementována serverová část aplikace. Také byly použity pro realizaci systému open source serverové aplikace běžící v prostředí linux, a to syslog-ng, honeyd, snort.

Celý systém jsem nazval symbolicky Controleye, což lze interpretovat jako řídící nebo kontrolní (dohlížející) oko. Název symbolizuje vlastnosti monitorovacího systému, pomocí něhož může s mírnou nadsázkou jeden správce monitorovat jedním okem (pohledem) celou počítačovou síť.

Aplikaci pro sběr dat jsem nazval Controleye – Agent a serverovou část aplikace jsem nazval Controleye – dashboard. Serverová část aplikace může běžet jak na internetu, tak v lokální síti na webovém serveru s podporou PHP a dostupnou databází MYSQL. Já pro odzkoušení internetové komunikace zaregistroval doménu www.controleye.cz, na níž jsem spustil serverovou část aplikace pro firmy, co nemají vlastní server. Do aplikace mají uživatelé přístup prostřednictvím uživatelského jména a hesla.

Sběr dat na síťových prvcích (vyjma počítačů s OS Windows) zajišťuje syslogový server syslog-ng, ten vše ukládá na lokální disk, z kterého se pomocí parserů (skriptů psaných v php a spouštěných v pravidelných intervalech) vše transformuje do databáze. Nad touto databází pak už je realizována serverová část aplikace nazvaná Controleye - dashboard .

To s jakých modulů je síť sestavena je vidět na obrázku níže. Červenou línii je označeno rozhraní, které může být a nemusí být v systému zavedeno. Jde o rozhraní mezi firemní a internetovou sítí. V případě kdy firma vlastní svůj webový server je možné vynechat tuto červenou hranici systému a vše nasadit v rámci firmy jako jeden celek.



Obrázek 7 - Schéma znázorňující tok logových dat z logovaných zařízení ke správci systému

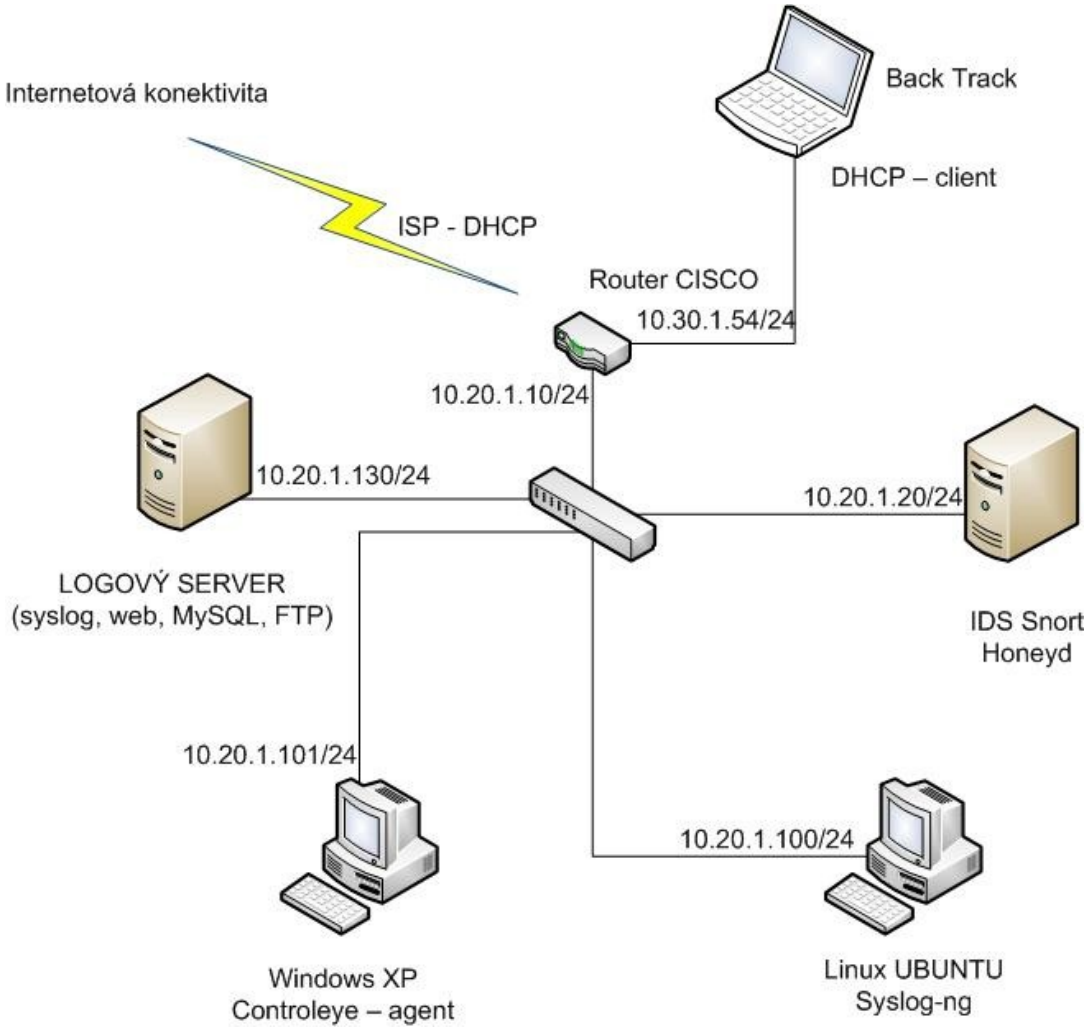
8.1 Testovací prostředí

Síť, pro kterou jsem systém stavěl, jsem si vytvořil ve virtuálním prostředí, ať je možné simulovat provoz a útoky aniž by došlo ke komplikacím na reálné síťové infrastruktuře. Použil jsem vizualizačního nástroje (aplikace) GNS3 a programu pro vizualizaci počítačů VirtualBox. Nastavil jsem si síť, která měla vstupní prvek do firemní sítě (router). Za tímto prvkem pak dále běžela firemní síť symbolizovaná dvěma virtuálními počítači. Jeden počítač se systémem Windows XP a druhý počítač se systémem Linux s distribucí Ubuntu. Všechen provoz v této síti byl monitorován sondou Snortu. Aplikace Snortu běžela na vizualizovaném PC s OS Linux Ubuntu. Na tomto stroji jsem kromě Snortu nainstaloval i deamona Honeyd. Ten měl za úkol zmást útočníky a jejich útoky. V testovací síti byly dále umístěny dva počítače pro uživatele. Jeden byl s OS Windows a byl na něm

nainstalován AGENT systému Controleye, který přenášel všechny data protokolem FTP do souborového úložiště na Logový server. Druhý počítač měl nainstalovaný OS Linux a logování událostí na tomto PC bylo zaznamenáváno pomocí syslog-ng démona a následně posíláno prostřednictvím protokolu syslog na logový server.

Pro testování jsem do sítě připojil i Linuxovou stanici s distribucí BackTrack, kterou jsem se snažil simulovat útočící PC. To je v jiném síťovém segmentu z důvodu reálného testovacího provozu.

Celá testovací (virtuální) síť vypadá, jak je znázorněno na obrázku:



Obrázek 8 - Virtualizace testovací sítě

V dalších kapitolách se budu věnovat konfiguraci a vytvoření jednotlivých částí systému až po centrální pult správce, který má pro sledování sítě k dispozici. Po jednotlivých částech se pokusím vysvětlit cestu logované události před oči správce, tak aby přišla v přehledné formě.

8.2 Controleye – Agent

Controleye - Agent je aplikace, která má za cíl monitorovat co nejvíc možných událostí na počítači s operačním systémem Windows. Aby aplikace byla užitečná, musí být aktivní po celou dobu provozu počítače. To znamená i ve chvíli, kdy na počítači není přihlášen žádný uživatel. Aplikace je proto koncipována a spouštěna jako služba. Tím je zajištěn monitoring stanice od chvíle, kdy uživatel stiskne tlačítko zapnout.

Jak již bylo obecně uvedeno výše, aplikace je napsaná v jazyce C# nad platformou .NET, proto je pro její provoz potřeba mít nainstalován i framework, nejméně ve verzi 2. Protože jde o velmi malou samostatně spustitelnou aplikaci, instalační rutina není potřebná, proto je potřebné aplikaci na všechny počítače pouze nakopírovat a nastavit pro každou stanici zvlášť dle potřeb. Do budoucna vidím však možnost aplikaci nasazovat v síti také vzdáleně pomocí instalační rutiny, která by zjednodušila i tuto práci. Bylo by pak možné v jednom síťovém segmentu instalovat všechny počítače najednou.

Aplikace je nachystaná modulárně tak, aby bylo kdykoli možné dopsat další typy kontrol. Udržuje si však centrální koncepční myšlenku celého systému. To protože z předchozích zkušeností vím, že prakticky neexistuje verze aplikace, kterou lze prohlásit za definitivní, což vtipně vystihuje úsloví: „Hotová je jen ta aplikace, kterou už nikdo nepoužívá“.

V současné době jsem v rámci aplikace naprogramoval čtyři moduly. Ty monitorují disk, operační paměť, procesor a pro mě nejpodstatnější část - logovaní celého systému.

Z monitoringu disku jsou získávány informace o velikosti všech disků, jejich názvech i popiscích stejně tak, jako o velikosti zbývajícího volného prostoru a formátování.

Monitorování paměti a procesoru je velmi obdobné, protože mě zajímají zejména tři údaje. A to:

- maximální využití nebo vytížení paměti a procesoru,
- aktuální využití paměti a procesoru
- průměrné využití po dobu od posledního odeslání informací na centrální server.

U obou monitorovacích modulů si uživatel může prohlédnout aktuální stav hodnot, které jsou viditelné po překliknutí na daný modul kontrol.

Monitoring logu počítače je nejobsáhlejším modulem aplikace. Proto je také jako jediný modul konfigurovatelný. Jsou na něm nastavitelné možnosti zasílání zpráv na centrální server. A to pouze zprávy, které mají prioritu:

- warning (upozornění)
- error (chyba)

- information (informace)
- nebo jakákoli jejich kombinaci.

Systém funguje následujícím způsobem. Pokud je poprvé aplikace spuštěna, pak systém pošle na server kompletní log počítače, jenž byl vytvořen do té doby. Z toho důvodu první průběh kontroly trvá déle a i množství odeslaných dat je větší. Další kontroly však už nekontrolují celý log. Hledají v protokolech logu pouze novější informace, než byly poslány na centrální server.

V aplikaci je pak uživateli k dispozici panel globálního nastavení, v kterém správce může nastavit identifikátor monitorované stanice, přihlašovací jméno a heslo pro vytvoření FTP spojení s centrálním serverem. Nemůže nastavit v aplikaci jméno hostitele, protože by bylo pak možné zneužít posílání dat na jiný server, který není součástí systému Controleye. Poslední a přitom podstatnou věcí, kterou správce musí nastavit, je časový interval, v jakém bude probíhat kontrola na jednotlivých počítačích. Čas se nastavuje v minutách a je možné jej nastavit jak na velmi krátký, tak na velmi dlouhý interval podle toho, jak správce uzná za vhodné. Na základě vlastních zkušeností doporučuji používat interval 5 až 15 minut, který není zas až tak krátký, aby zbytečně vytěžoval síťový provoz, a přitom nabízí správci dostatečně včas potřebné aktuální informace.

Jak lze vyčíst z návrhu systému v předchozí kapitole, informace z agenta odcházejí prostřednictvím protokolu FTP ve formátu XML. Ve výhledu je také dopracování komunikace i prostřednictvím protokolu HTTP. Důvodem proč i protokol HTTP je fakt, že v některých firmách jsou nastaveny firewally i pro odchozí provoz, takže protokol FTP může být blokován, což nastoluje další problém konfigurace či nastavení výjimek. U protokolu HTTP je tato pravděpodobnost menší.

Formát XML jsem zvolil z důvodu, že zpráva cestuje ve formátu, který drží strukturu přenášených dat a zároveň dovoluje mít dostatečně zabezpečený centrální server, na který se informace sbíhají. Možnost, že by se informace na server nahrávaly přímo do databáze, jsem nevyužil vzhledem k nebezpečí možného hackerského útoku na databázový server. Takový útok je jen velmi těžké spolehlivě ošetřit, pokud by měla být databáze otevřená pro komunikaci z venkovních sítí pro jakoukoli IP adresu. Dalším důvodem je, že pokud obsluhu vkládání dat do databáze bude řešit jedna služba, pak není možné, aby došlo ke kolizi zápisu, jak by tomu mohlo být v případě zápisu velkého množství agentů.

Strukturovaný formát XML zprávy udržuje logiku aplikačních modulů, kdy každý z modulů má svůj vlastní uzel připravený tak, aby se data ze souboru jednoduše dostávala do databáze.

Krátkou ukázku generovaného XML uvádím níže. Ukázka prezentuje:

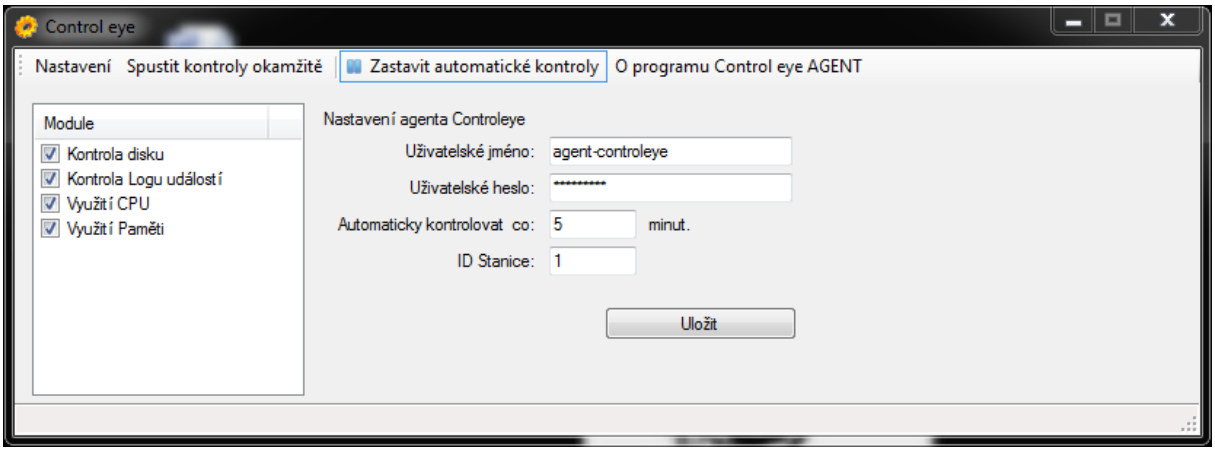
- kdy byla data pořízena,
- na jakém zařízení,
- že na zařízení byly spuštěny moduly pro kontrolu disku a logu.

Výpis modulů pro kontrolu disku a logu je z důvodu původní obsáhlosti a účelu přehledné ukázky krácen. Níže prezentovaná data jsou pořízena klientskou částí systému Controleye - Agent na reálném počítači.

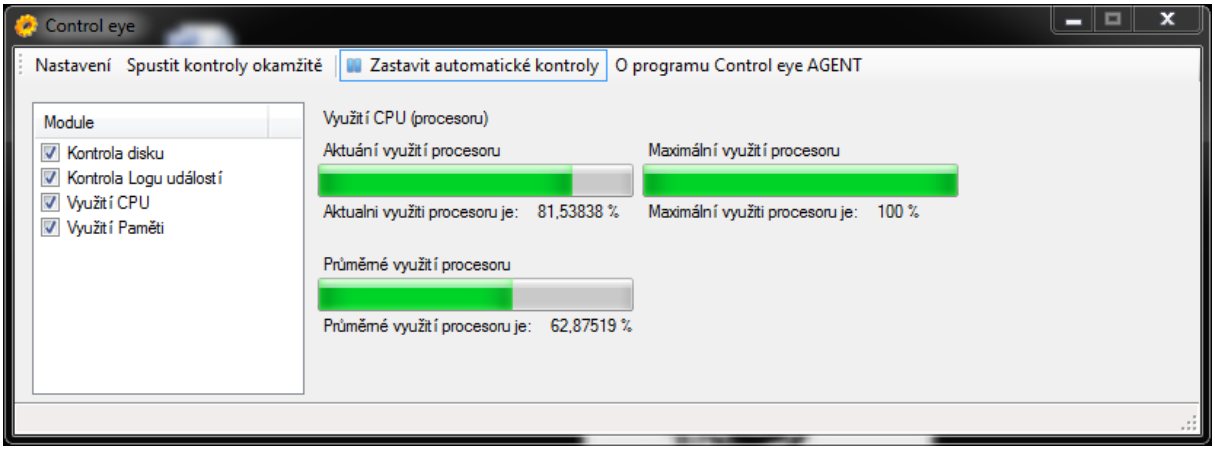
```
<ControlEyeResult Time="5.4.2012 13:10:03" deviceID="1">
  <ModuleResult Module="Kontrola disku">
       <DriveInfo Name="C:\" VolumeLabel="Windows7_OS" DriveType="Fixed" FileSystem="NTFS"
       FreeSpace="91912609792" FreeAvailableSpace="91912609792" TotalSize="308326428672" />
  </ModuleResult>
  <ModuleResult Module="Kontrola Logu událostí">
    <EventLogList Log="Application">
      <EventLogeEvent Date_Time_Generated="2012-04-05 11:43:03" Date_Time_Written="2012-04-05
11:43:03" Event_ID="0" Entry_Type="Information" Message="Nebyl nalezen popis objektu Event ID
0 ve zdroji gupdate. Místní počítač pravděpodobně neobsahuje potřebné informace registru či
soubory DLL zpráv určené k zobrazení zpráv nebo k nim nemáte oprávnění k přístupu. Následující
informace jsou součástí události:'Service stopped'" />

    </EventLogList>
       <EventLogList Log="Cisco AnyConnect VPN Client" />
       <EventLogList Log="HardwareEvents" />
       <EventLogList Log="Internet Explorer" />
    <EventLogList Log="Key Management Service" />
    <EventLogList Log="Lenovo-Message Center Plus/Admin" />
    <EventLogList Log="Media Center" />
    <EventLogList Log="ODiag" />
    <EventLogList Log="OSession" />
    <EventLogList Log="PRTG Network Monitor" />
    <EventLogList Log="Security" />
    <EventLogList Log="Windows PowerShell" />
  </ModuleResult>
</ControlEyeResult>
```

Celá aplikace pak vypadá takto:



Obrázek 9 Controleye- Agent



Obrázek 10 Controleye – Agent, kontrola CPU

8.3 Serverová část aplikace

Součástí této práce bylo hledání řešení, jak zajistit sběr informací o všech prvcích počítačové sítě tak, aby měly co nejlepší vypovídací hodnotu. Hledal jsem řešení, které by odpovídalo podmínkám:

- nulové náklady,
- maximální možná přizpůsobivost,
- jednoduchá konfigurovatelnost
- nasaditelnost na testovací síti.

Když jsem zjišťoval, jak lze monitorovat síťové prvky, které jsem měl k dispozici (přepínače a směrovače od společnosti CISCO) vyšly z toho dvě možnosti. Obojí má možnost být monitorováno prostřednictvím protokolů SNMP a Syslog. Z toho důvodu jsem pro účel své práce přidal další výběrovou podmínku, a to schopnost využívání protokolu Syslog, který se mi jeví jako jednodušší varianta pro sběr událostí.

Všem zmíněným požadavkům vyhovuje open-source řešení s názvem již výše popisovaným syslog-ng. Jedná se o serverovou aplikaci, která protokol Syslog vhodně používá pro sbírání a škatulkování logových zpráv.

Syslog-ng existuje jak ve verzi open-source, tak ve verzi komerční. Ta umožňuje navíc přímé ukládání do databáze. V případě nasazení komerční verze by se zjednodušil současně nastavený systém sběru dat o parsery, které musí sesbírané informace přenést z logových souborů do databáze.

8.3.1 Nastavení syslogového serveru syslog-ng pro sběr dat

Pro kumulaci logových informací jsem si zvolil právě výše popisovaný syslog-ng, který v open-source verzi se jevil jako nejvhodnější řešení sběru logových informací z jednotlivých síťových

prvků. Mohu jej instalovat jak na klientské stanice, kde se stará o odesílání logových informací, na logový server, tak i na servery kde zajišťuje službu logového serveru.

8.3.2 Nastavení serveru:

Syslogové servery defaultně naslouchají na portu 514 protokolu UDP. U mě je tomu taky tak. Proto, aby server přijímal všechny logy ze sítě, jsem nastavil.

```
source server1 {
  udp();
};
```

V části konfiguračního souboru označeného filtr jsem si vytvořil filtrovací pravidla pro zaznamenávání událostí, které bude server přijímat.

```
filter notdebug { level(info...emerg);};
filter f_snort    { match ("snort/[");};
```

Cíl logování jsem si ve virtuálním prostředí zvolil do několika souborů. V případě nasazení na reálné síti bych podle potřeby a běžících služeb bych ještě volil variantu, která by v sobě zahrnovala hierarchické zařazování do složek podle data, tak aby ve velkém množství dat bylo možné se rychle orientovat.

```
destination server_SNORT {file("/var/log/server/SNORT.log");};
destination server_ROUTERY {file("/var/log/server/ROUTERY.log");};
```

Proto, aby se události zaznamenávali, jsem vše výše zmíněné spojil pomocí pravidla, kterým démonovi říkám co, jak a kdy (za jakých podmínek) budu logovat.

```
log {source(server1);filter(f_snort);destination(server_SNORT);};
log {source(server1);filter(notdebug);destination(server_ROUTERY);};
```

Pravidel můžeme mít, kolik potřebujeme. V případě reálné sítě bych si oddělil do různých souborů události podle významnosti. To ve virtuální síti nedělám, protože vše testuji na vlastním notebooku, který se spuštěním celé virtuální sítě má problémy. Proto se snažím minimalizovat množství aplikací běžících současně. Tím, že se ve virtuální síti vše sbírá do jednoho souboru, mohu spouštět pouze jen jeden parser, který mi v logové souboru vybere všechny potřebné informace a ty zaznamená do databáze, tím šetřím nároky na výkon. V případě větší sítě by se pak dalo nastavit několik typů parserů, které by získávali data z různých logový souborů (rozdělených třeba podle významu události) a mohly by být spouštěny v různých intervalech, tak aby události s vysokou prioritou byly co nejdřív před očima správců, zatímco nedůležité informace nezatěžovali tolik chod serveru.

Na jednotlivých Linuxových stanicích se pak musí nastavit nasměrování na logový server, které se provede vytvořením pravidla. V něm říkám, že cíl logových zpráv má být na vzdáleném logovém serveru:

```
destination from-server1 {
  udp("10.143.0.1");
};
```

Samozřejmě vše jde opět nastavit tak, aby se logovalo jak na lokální stanici, tak i na server. Což je dle mého názoru vhodná varianta. Vzniká tak záloha logových informací v případě, kdyby selhala nějaká část logovacího systému.

Tímto nastavením se nám podařilo dostat všechny logované události na síti na jeden logový server. Máme je uloženy v jednom logovém souboru a v jedné složce.

V logovém souboru se nám sbíhají události ze všech síťových prvků, na kterých běží syslog. Ve složce se nám sbíhají logované informace ze všech pracovních stanic, na nichž běží OS Windows a je na nich nainstalovaný Controleye – agent.

Do těchto dvou míst přistupuje aplikace Controleye – dashboard pomocí parserů v pravidelných intervalech a nahrává získané události do databáze. Parserem můžeme v mém případě myslet php skript, který je v pravidelném pětiminutovém intervalu pouštěn. Systém Controleye – dashboard je nachystaný tak, aby bylo možné jednotlivých parserů přidávat dle potřeby a velikosti sítě - množství monitorovaných služeb na síti. O controleye – dashboard píšu níže.

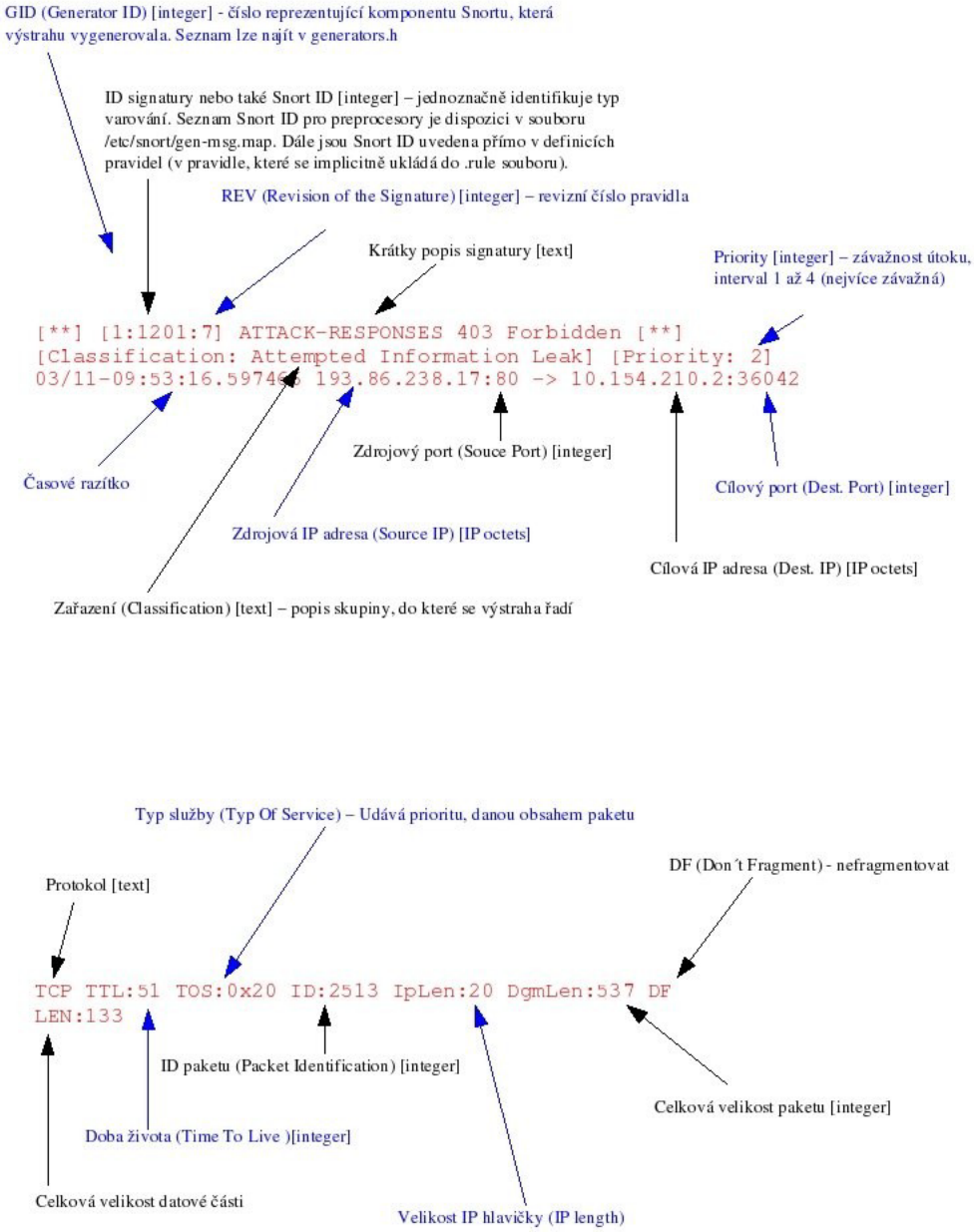
V mnou vytvořené virtuální síti jsem spustil služby aplikací SNORT na pozici NIDS, aplikaci Honeyd v roli hoeypotu. Dalšími událostmi, které monitoruji, jsou síťové přepínače a směrovače. Z nich jsem získával informace o stavu sítě.

SNORT a HONEYD běží v rámci úsporných výkonových opatření na jedné virtuální stanici. Z té Syslog-ng odesílá informace na centrální logový server.

Logované události z jednotlivých služeb zkusím trochu rozebrat.

8.4 Logy ze Snortu

Pro popis logovaných událostí jsem si dovolil vzít velmi výstižný obrázek z práce Radomíra Orkáče, kterou vypracoval v rámci projektu do předmětu Směrované a přepínané sítě na Vysoké škole Báňské – Technické univerzity Ostrava v roce 2006. Výstraha je vybraná ze souboru *„/var/log/snort/alert“* a na následujícím obrázku vysvětlena.



Obrázek 11 Vysvětlení logové alert zprávy Snortu

8.5 Logy z Honeydu

Honeyd vytváří log pro všechny spojení a pakety směřované na virtuální síť.

Záznamy mohou vypadat třeba takto:

```
2012-05-07-16:48:30.1212 tcp(6) S 192.168.1.135 33395 10.3.0.1 22
[Linux 2.6 ]
2012-05-07-16:48:41.4929 tcp(6) S 192.168.1.67 22110 10.3.0.11 21
[Windows XP SP1]
```

První pole obsahuje datum a čas, kdy se událost stala na setiny sekundy

Druhé pole obsahuje protokol, kterým bylo komunikováno (**tcp**, **udp**, or **icmp**).

Třetí pole může obsahovat písmena **S** – indikuje start nového připojení, **E** – indikuje konec připojení, nebo když nepatří do žádného spojení.

Další čtyři pole představují spojení čtyři skupin: <src ip, src port, dst ip, DST port>.

Pro pakety TCP, které nejsou součástí spojení, Honeyd zaznamená velikost paketu a TCP příznaky za dvojtečkou.

Komentáře, jako je třeba identifikace operačního systému, jsou připojeny na konec řádku.

8.6 Logy z routeru

Aby bylo možné přijímat logy ze směrovače Cisco bylo třeba na směrovači spustit logování pomocí protokolu syslog.

K tomu postačila tato série příkazů:

1. Router# **configure terminal -** vstup do módu konfigurace.
2. Router(config)# **service timestamps** *log* - nastavení typu logování
3. Router(config)#**logging** *10.20.1.130* - adresa logového serveru
4. Router(config)# **logging trap** *6* - úroveň logování (6 – Information)
5. Router(config)# **logging facility** *local0* - nastavení označení zařízení

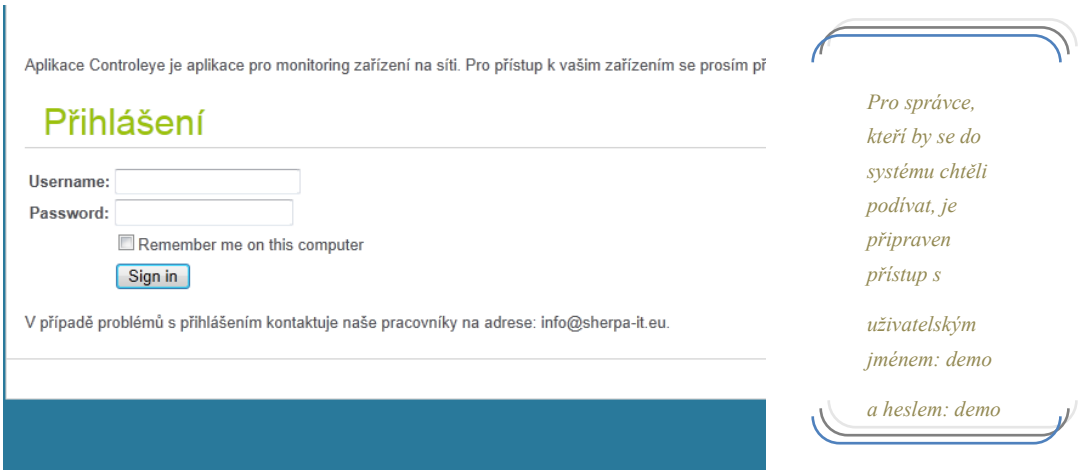
Na logovém serveru se nám pak objevili zprávy, které zaznamenával směrovač. Všechny zprávy z tohoto zařízení nesou označení local0, takže je jednoduše můžeme rozpoznat a začlenit do systému controleye.

8.7 Controleye – Centrální panel (dashboard)

Centrální panel by se dal nazvat řídícím pultem celého systému. Sbíhají se do něj všechny informace z klientských stanic, stejně tak jako ze všech aktivních síťových prvků. Sběr dat probíhá, tak jak bylo popsáno v předchozích kapitolách. Na centrálním panelu pak probíhá agregace jednotlivých získaných informací tak, aby správce měl co nejlehčí diagnostiku problému a v případě potřeby mohl do chodu sítě zasáhnout patřičným opatřením.

Aplikace je psaná v programovacím jazyce PHP nad frameworkem Nete. Proto, aby byla dostupná odkudkoli, běží na webovém serveru, který je k dispozici jak v lokální síti (v případě firmy s vlastním serverem), tak celosvětově na síti internet, protože má veřejnou IP adresu. Pro server jsem zaregistroval doménu s názvem, který zaštiťuje celý systém, a to controleye.cz

Protože je aplikace veřejně dostupná, bylo třeba zajistit, aby k získaným datům měl přístup jen pověřený správce celého systému. Proto při vstupu do aplikace prostřednictvím internetového prohlížeče systém vyžaduje zadání uživatelského jména a hesla.



Obrázek 12 - Přihlášení uživatele

V současné době je aplikace navržena tak, že umožňuje přístup všem přihlášeným uživatelům na stejné úrovni. Není tedy rozdílu v přístupových právech do jednotlivých částí aplikace. Aplikace je tedy v současné době navržena pro správu i více síti jediným správcem, resp. i více správci, kteří spravují společné zákaznické sítě. Tato varianta plně odpovídá jak mým praktickým potřebám, tak i zadání vlastní diplomové práce. Současně uvedené řešení umožňuje další rozvoj systému doprogramováním víceúrovňového a víceuživatelského (je možné rozlišit přístupu do aplikace podle zákazníku a jejich sítí), tak aby bylo možné služby této aplikace nabídnout i jiným firmám, které mají na starost jimi spravované zákaznické sítě.

Po zadání přístupových údajů je uživatel po dobu aktivního pohybu v aplikaci přihlášen, a to do doby, než se z aplikace odhlásí. Pokud by nedošlo k odhlášení z aplikace, systém vyhodnotí nečinnost sezení (session) uživatele a daného správce ze systému odhlásí po nastaveném čase automaticky sám.

Ve chvíli, kdy je uživatel do systému přihlášen, má před sebou centrální panel, který je rozdělen do několika logických částí. Jednotlivé bloky můžeme nazvat podle významu:

- hlavičkou centrálního panelu,
- navigačním stromem sítí,
- a detailem.

8.7.1 Hlavička centrálního panelu

V hlavičce centrálního panelu systému jsou pouze informace o přihlášeném uživateli a názvu firmy, do které uživatel patří. Je zde umístěno mé logo a odkaz pro odhlášení ze systému.

8.7.2 Navigační strom centrálního panelu

Obrázek 13 - Navigační strom a hlavička centrálního panelu

Popisovaný systém rozlišuje hierarchické zařazení sledovaných zařízení ve třech úrovních:

1. Monitorovaní zákazníci
2. Skupiny monitorovaných zařízení
3. Monitorovaná zařízení.

V navigačním stromu je zobrazena struktura prvních dvou úrovní, z nichž jsou patrní jednotliví monitorovaní zákazníci a jim příslušné skupiny monitorovaných zařízení. Skupin monitorovaných zařízení může mít zákazník, jenž požaduje monitoring svého zařízení, nekonečné množství. Skupinou monitorovaných zařízení v tomto případě není skupina počítačů a aktivních síťových prvků, které by měly společnou síť (myšleno z hlediska adresování sítě prostřednictvím protokolu TCP/IP). Skupinou monitorovaných zařízení je v tomto pojetí množina zařízení, které mají pro správce příp. i zákazníka logickou souvislost. Proto je možné vytvářet skupiny, jako v mém případě, na testované síti podle umístění či způsobu využití počítače nebo typu zařízení.

8.7.3 Detaily centrálního panelu

Nejobsáhlejší částí systému je a pravděpodobně vždy zůstane detail monitorování. V detailu se totiž sbíhají informace o monitoringu celé zákaznické sítě. V detailu se vždy zobrazuje informace související s vybranou částí v hierarchické stromové struktuře zákazníků a jejich skupin. V současné podobě má detail tři typy zobrazení. (Řešení je připraveno na další rozšíření. Záleží jen na požadavcích zákazníků, zda budou vyžadovat rozšíření funkcionality systému. Pokud by tomu tak bylo, určitě se vyplatí systém rozvíjet dál.)

Prvním detailním pohledem do systému je pohled na monitorovaného zákazníka. Pokud správce ve stromové struktuře vybere jakéhokoli zákazníka, jenž si monitoring objednal, zobrazí se mu detailní informace o stavu celé sítě. Jedná se o velmi sofistikované řešení, kdy správci systému jsou předkládány jednoduché informace v podobě agregovaného logu. Jde tedy o tabulkové zobrazení informací, tak jak je zná každý správce z kteréhokoli operačního systému nebo jen logového souboru, kdy je vytvořena událost, která má datum, čas a vlastní agregovanou zprávu. Vzhledem k potřebě maximální přehlednosti není vhodné ke zprávě přiřazovat další údaje, jako je např. zdroj původu. Systém sleduje zákazníka jako celek, který by se dal připodobnit k jednomu velkému operačnímu systému. Pokud je ale vytvořena v systému nějaké zpráva, která je zaznamenána k jednotlivému

zákazníkovi, vždy se jedná o agregovanou zprávu z několika zařízení, nebo několika typů událostí z jednoho zdroje příp. jejich kombinace.

Protože taková zpráva vždy nemusí být dostačující, má správce po rozkliknutí dané zprávy možnost se podívat z jakých logových záznamů vznikla tato agregovaná událost až do úrovně zdroje zprávy.

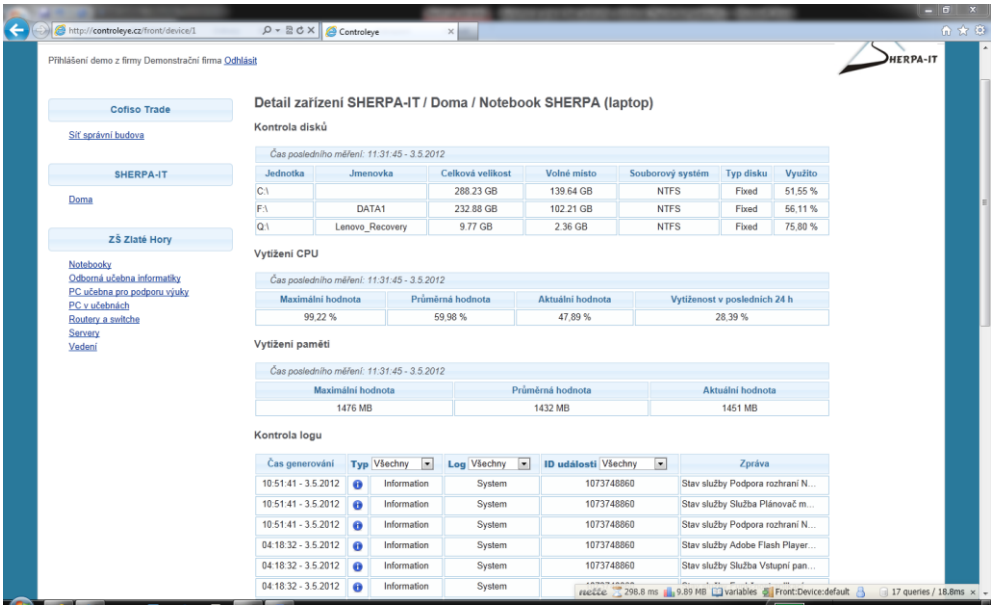
To, jak taková agregovaná zpráva vzniká, je popsáno v následující kapitole diplomové práce, věnované logovým souborům a logovým informacím. Dalším detailem, který je správcům k dispozici, je detail skupiny monitorovaných zařízení. Tento detail nabízí správcům ucelený přehled na všechna zařízení v dané skupině. Jednotlivá zařízení jsou zobrazena opět tabulkovým výpisem, přičemž pro každé zařízení je veden jeden záznam tabulky, ve kterém správce nalezne nejpodstatnější informace o daném zařízení. Mezi nejpodstatnější informace rozhodně patří poslední kontrola daného zařízení. Je to čas, kdy zařízení naposledy poslalo informace o svém stavu do centrálního panelu. Pak následují pole, která dávají přehled o součtech jednotlivých zaznamenaných logovaných zpráv podle typu priority.

Následuje pole s informací o množství hackerských útoků provedených na stanice s OS Windows. Množství hackerských útoků se počítá jako počet zpráv zaznamenaných do logu s neúspěšným pokusem o přihlášení. Nejedná se jen o neúspěšná přihlášení ale o všechny zprávy s ID dle logu MS Windows: 529, 530, 531, 532, 533, 534, 535, 539, 548, 675, 676, 672, 644, 4625. Ve chvíli, kdy překročí množství zpráv s jakýmkoli z těchto ID (i součtem všech) za období dvaceti čtyř hodin hodnotu 100, systém vyhodnotí, že na dané zařízení byl proveden hackerský útok, agregované hlášení je zvýrazněno a správce by měl danému počítači věnovat větší péči, tak aby ke stejné situaci nedocházelo.

Posledním neméně důležitým polem, které se nemění, ale má vliv na zobrazované informace je typ zařízení. Pokud jde o zařízení typu PC, je zobrazeno větší množství informací v posledním detailu daného zařízení, než u zařízení, které má ve v poli typ uvedeno router nebo switch. U těchto zařízení (router, switch) se sbírá menší množství informací, než u zařízení typu PC (laptop, workstation).

Jak jsem naznačil o odstavec výše, posledním, v současné době implementovaným typem detailu, je přímo detail monitorovaného zařízení. Jedná se o přehlednou sumarizaci informací získaných na daném zařízení. V případě počítače jsou v posledním detailu zobrazeny aktuální informace o stavu zařízení monitorovaných Controleye –Agentem.

Jak již bylo popsáno ve stati o Controleye -Agentu, jsou zde prezentovány informace o stavu disků, jejich kapacitách a volném místě. Vše v přehledné tabulce. Zároveň je administrátor upozorněn zvýrazněným polem ve chvíli, kdy u daného zařízení hrozí brzské zaplnění disku. Takové zvýrazněné upozorňuje na problém, který by měl správce sítě řešit tak, aby došlo k co nerychlejší nápravě.



Obrázek 14 Controleye - centrální panel, detail zařízení

Další přehlednou informaci, která má správce díky systému k dispozici, je informace o stavu CPU (procesoru). Vždy je zobrazena hodnota z posledních měření. Hodnoty jsou tři, a to maximální, průměrné a aktuální vytížení procesoru za dobu monitorovacího intervalu. Agregovanou informaci, kterou správce dostává, je navíc průměrná hodnota využití procesoru během posledních 24 hodin. Je to hodnota spočítaná ze všech naměřených průměrných hodnot za období 24 hodin. Pokud je tato hodnota nad 80 %, pole je opět zvýrazněno, aby mu správce věnoval větší pozornost. Důvodem je fakt, že počítač, který má vytížený procesor nad tuto průměrnou hodnotu, bude nejspíš mít buď zastaralý hardwarově, nebo bude vykazovat vysokou pravděpodobnost zavirování.

Stejnou informaci jako o procesoru, najdeme v tomto detailu i informaci o paměti monitorovaného počítače. Opět jsou zde zobrazeny tři informace: aktuální, maximální a průměrná hodnota.

Poslední částí detailního přehledu tvoří pro správce ten nejpodstatnější zdroj informací o dění na PC. Jsou to záznamy z logových souborů, které se zobrazují opět v přehledné tabulce, tak aby správce viděl, kdy byla událost zaznamenána, s jakou prioritou byla zaznamenána a v kterém protokolu byla zaznamenána. Logové zprávy jsou v tabulce zobrazovány od nejaktuálnějších k těm, které už jsou z časového pohledu starší, protože množství logových zpráv bývá velké, i když se jedná jen o jeden počítač. Jedná se o počty záznamů v řádech desítek tisíc za rok běhu. Proto jsou pro správce nachystána pole, v nichž zprávy může vyfiltrovat podle priority, ID zprávy nebo typu protokolu, do kterého zpráva byla zaznamenána. Další možností, kterou systém nabízí, aby se v logu správce orientoval, je stránkování, které najde vždy na konci právě zobrazeného seznamu zpráv.

9 Události na síti a jejich agregace

9.1 Stručný popis implementovaného řešení agregace dat

Realizované a implementované řešení agreguje data v několika krocích, a to dvěma způsoby. Controleye sbírá jednak data z log souborů resp. protokolů síťových stanic a jednak data různých logů dalších aktivních síťových prvků. Detailní popis problematiky logů je uveden v následující podkapitole.

Controleye - Agent, instalovaný na PC, ověří změny logů příslušné stanice oproti předchozí kontrole, změny transformuje do formátu XML a odešle na centrální server. Zde jsou data uložena do relační centrální databáze s pevnou datovou strukturou. Ostatní aktivní prvky odesílají data filtrovaná dle přednastavené priority na Syslog server, jenž data následně může dále filtrovat a výsledek sběru ukládá do jediného souboru, jenž je v pravidelných intervalech ověřován na změny, a dosud neevidované výsledky jsou načteny do centrální databáze. V prvním stupni agregace jsou takto shromážděna a transformována původně nekompatibilní data heterogenních struktur ze všech aktivních prvků sítě do jediné databáze s jednotnou datovou strukturou.

Nad takto shromážděnými daty jsou provedeny agregace druhého stupně. Jedná se o:

- vyhodnocení hackerských útoků,
- vyhodnocení přístupu na webové stránky z jednotlivých stanic,
- agregaci zpráv ze všech aktivních prvků o připojení jednotlivých stanic do sítě,
- sumarizace událostí dle společné priority.

Agregovaná data jsou přehledně prezentována v centrálním panelu systému Controleye se zvýrazněním nejdůležitějších varování. Uvedená problematika agregace je detailněji popsána v následujících kapitolách.

9.2 Agregace logu

Jak již bylo uvedeno v předchozí kapitole, není agregace logů vždy jednoduchou záležitostí. U počítačových sítí, které jsou navrhovány jednotnou koncepcí, nejlépe s prvky téhož výrobce (jako např. již popsané řešení CISCO MARS), lze počítat s využitím jednotných identifikátorů událostí a tedy relativně jednoduchým způsobem zpracování a agregace takto vytvářených zdrojových dat.

Systémovým návrhářům však vzniká velký problém ve chvíli, kdy chtějí vytvořit sofistikovaný systém, který by uměl agregovat logové informace z více heterogenních zdrojů.

Ve své diplomové práci jsem se problematikou agregace heterogenních dat zabýval a došel jsem k závěru, že není snadné najít obecné řešení, které by platilo pro všechny sítě. Agregovat

informace z jednotlivých zdrojů je možné jen v obecné a omezené míře. Není reálné hledat související informace, z různých síťových prvků od různých výrobců ve chvíli, kdy chybí informace, které by spojovaly a jednoznačně identifikovaly sledované události.

Mé řešení vychází z vlastního návrhu a implementace navrženého agregačního systému nad značně nesourodou sítí, pro niž bylo nutno hledat konkrétní řešení, které by platilo právě pro tuto síť. Je škoda, že neexistuje jasná norma, která by logové zprávy sjednotila. Daly by se pak nad takovýmito systémy vytvářet obecně funkční aplikace, které by správcům zjednodušily práci. Je zřejmé, že ve chvíli, kdy by bylo možné hledat společné prvky v logových zprávách na zařízeních od různých výrobců stejně jednoznačně, bylo by možné podstatně lépe spojovat logové zprávy do agregačních skupin.

Vzhledem k nesourodosti hardware a rozdílnosti struktury jím logovaných dat, bylo nutno vytvořit agregační skupiny na bázi společných znaků logovaných dat, které by umožnily správcům prezentovat události jednodušším způsobem.

V rámci jednoho zařízení, takových možností agregací zpráv můžeme najít relativně dost. Avšak mezi nesourodými prvky je seskupování značně problematičtější. Nejčastějším spojovacím prvkem, který může pomoci při hledání souvislostí a vazeb, je čas. Podle času a porovnání obsahu může člověk jednotlivé zprávy seskupit. Avšak počítačová logika takové porovnání obsahu jednoduše neumožňuje. Dalším problémem je, že čas není jednoznačným identifikátorem, který by zaručoval souvislost mezi zprávami, jelikož není zaručena nejen přesná časová souslednost tvorby událostí, ale ani nastavení času na jednotlivých objektech sítě.

Ve zprávách z logů OS Windows je identifikace jednoznačně určena ID události. Ostatní síťové prvky své ID zpráv sice mají, ale tato ID vůbec nesouvisí s identifikátory z jiného typu logu. Společným rysem, jehož lze někdy využít, je síťová spojitost vazby IP a MAC adres. Tu se mi v několika případech podařilo najít a při řešení využít.

Vzhledem k tomu, že síť, na které vytvářím a testuji navržený systém virtuální a dobře zabezpečená, povedlo se mi převážně zachytit a agregovat spíše zprávy ze simulovaného přenosu dat. Výtah ze souvisejících událostí uvádím v následujících odstavcích.

9.2.1 Agregace informací o hackerských útocích

Při procházení studovaných systémů jsem zaregistroval, že jeden ze systémů vyhodnocuje tato ID Windowsových zpráv: 529, 530, 531, 532, 533, 534, 535, 539, 548, 675, 676, 672, 644, 4625, 1073742836 pod jednou agregovanou událostí. Ve všech těchto případech jde o zprávy, kdy se jedná o narušení operačního systému. Za takové narušení systému je například považována zpráva zachycená mnou vytvořeným systémem Controleye s ID 1073742836 (zpráva: *Vzdálená relace z klienta s názvem a překročila maximální povolený počet neúspěšných pokusů o přihlášení. Relace byla nuceně ukončena.).* Zpráva zaznamenává větší počet chybných přihlášení do systému. Pokud je

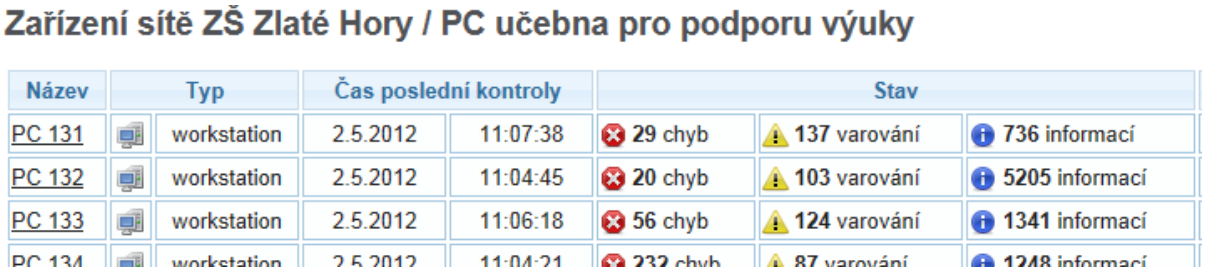
těchto událostí zaznamenáno během měřeného času určitý počet, lze předpokládat, že došlo k útoku. Podle toho o jaký systém jde, je třeba nastavit úroveň počtu zpráv, který bude vyhodnocen už jako útok. V mém případě je v systému Controleye nastavena hranice 100 událostí jakéhokoli typu. Jedná se tedy o agregaci logu v rámci jednoho zařízení.

9.2.2 Agregace přístupu na stránky z jednotlivých počítačových stanic

Agregovanou skupinou, která se zobrazuje jako jedna sumarizovaná zpráva je skupina logových zpráv vygenerovaných routerem. Jejím obsahem je součet všech navštívených adres z jednotlivých počítačů během jednoho dne. Pokud v součtu všech navštívených stránek je překročen limit 100MB na jedno zařízení je tato agregovaná zpráva zvýrazněna, tak aby uživatel zkontroloval, zda šlo o informace, které uživatel potřebuje k práci, nebo zda nezatěžuje zbytečně síť a konektivitu monitorované sítě k internetu. Zprávy tohoto typu se zobrazují v přehledu událostí pro danou síť. Pokud danou zprávu chce správce vidět, musí si zobrazit protokol daného zákazníka (v navigačním stromu kliknout na název zákazníka).

9.2.3 Agregace zpráv dle priorit

Agregace zpráv dle priority sčítá v databázi všechny události, které jednotlivá zařízení vygenerují, a v přehledu zařízení systém zobrazuje správci počítačové sítě informaci o počtu zaznamenaných událostí seskupených dle priority daných zpráv.



Obrázek 15 Sumarizace událostí dle typu priority.

9.2.4 Agregace zpráv ze všech aktivních prvků o připojení jednotlivých stanic do sítě

Jedná se o přehlednou sumarizaci všech zpráv, ze všech prvků v síťové topologii. Pod hlavičkou zprávy ve znění: „*Bylo připojeno zařízení na portu (označení portu) a switchi (označení switche)*“, se ukrývá skupina zpráv, které souvisí s aktivací portu. Jsou jimi převážně zprávy ze všech síťových prvků o změně topologie spanning tree.

Stejně tak, jak se zobrazuje zpráva o připojení portu a změně topologie, tak se zobrazuje obdobná zpráva i ohledně odpojení, rozdíl je jen ve slově *„připojeno“*, které je nahrazeno slovem *„odpojeno“*.

Protože se jedná o událost, která má společné jmenovatele přes celou síťovou strukturu, najdeme její obsah stejně jako u *„Agregace přístupu na stránky z jednotlivých počítačových stanic“* v protokolu událostí vedeného pro daného zákazníka.

V případě rozšiřování systému je možné hledat takových průniků různých událostí více. Je však racionální hledat zprávy, které mají pro správce sítě nějaký důležitější význam. Není např. efektivní předkládat prvoplánově agregace zpráv o bezporuchovém chodu sítě, když postrádají schopnost usnadňovaní práce správcům sítě.

9.2.5 Agregace zpráv z HONEYDU a SNORTU o útocích na síť

V případě, že je podezření na aktivitu, která v síti není očekávána, SNORT zareaguje a vytvoří událost. Upozornění na aktivitu uživatele (hackera). Ve chvíli, kdy jde o událost, kterou se hacker snaží dostat k zařízení, které je simulováno systémem honeyd vytvoří událost i deamon Honeyd. V případě kdy se tyto dvě události jak ze systému honeyd a snort shodují. Jsou systémem controleye vyhodnoceny a na dashboardu prezentovány jako jedna událost. Pod názvem „Scanování virtuální sítě honeyd, podezření na útok.“

Celá tato zpráva je založena třeba na složení těchto dvou zpráv:

Honeyd:

```
2012-05-07-16:48:30.1212 tcp(6) S 192.168.1.135 33395 10.3.0.1 22
[Linux 2.6 ]
```

SNORT:

```
[**] [1:483:2] ICMP Destination Unreachable Port Unreachable [**]
[Classification: Misc activity] [Priority: 3]
05/07-16:48:02.671446 192.168.1.135 -> 10.3.0.1
ICMP TTL:90 TOS:0x0 ID:2670 IpLen:20 DgmLen:92
Type:8 Code:0 ID:512 Seq:59153 ECHO
[Xref => http://www.whitehats.com/info/IDS154]
```

Systém detekuje shodu v IP adresách a čase. Když je čas v rámci dvou minut stejný a IP adresy taktéž, může považovat tyto zprávy jako související, proto je v rámci systému sloučí do jedné zprávy. V systému těchto pravidel může a bude narůstat, aby se k správci dostalo co nejvíce přehledných informací.

Někdo rozumný mi kdysi řekl, že informační systém, který je hotový, už nefunguje. Myslím si, že je to pravda a proto tento systém hotový není a mám v plánu jej stále dál rozvíjet, aby mi přinesl víc než jen zkušenost z prostudovaných aplikací. Třeba užitek při nasazení na komerční síti.

10 Testy systému Controleye

Testy systému Controleye probíhají nepřetržitě od doby vytvoření virtuální sítě a nasazení technik pro sběr událostí. Vzhledem k tomu, že jde o systém, který bude pravděpodobně dlouhodobě upgradován, popisuji stav, který byl zjištěn a pozorován od doby spuštění po současnost.

Controleye – Agenta se mi povedlo nainstalovat bez větších komplikací. Na síťových prvcích jsem standardním způsobem pomocí webového rozhraní nakonfiguroval vše potřebné, aby docházelo k logování aktivity síťových prvků.

Během zkušebního provozu jsem z logových informací sbíraných v centrálním panelu zjistil, že jsem měl na síti chybně nastavenou bezpečnostní politiku mezi jednotlivými sítěmi, proto nedocházelo k synchronizaci času na síťových prvcích s NTP serverem na síti. To byl první pozitivní přínos systému, který jsem ani neočekával.

Dalším poznatkem z testovacího provozu bylo zjištění, že je stanice neustále útočeno přes port vzdálené plochy. Denně došlo k několika tisícům pokusů o přihlášení prostřednictvím vzdálené plochy, proto jsem zvýšil zabezpečení přihlašování ke vzdálené ploše a problém jsem tím odstranil.

Ačkoli je systém Controleye dosud ve vývoji a testovacím provozu, přesto jej již monitoruji, a jsem přesvědčen, že díky němu určitě narazím na další varování, která mi umožní nalézt a napravit nedostatky. V současné době ale virtualní síť nevykazuje větších problémů a vše pracuje bez závad.

Již z počátečních výsledků je zřejmé, že systém Controleye přináší značný užitek a může plnit funkci dohledového centra v tom smyslu v jakém ji započal.

11Závěr

Práce popisuje současná řešení sběru a agregace dat událostí zaznamenaných různými zdroji, následně je analyzuje a na základě vyhodnocení navrhuje vlastní řešení. Dle návrhu byl zhotoven systém pro agregaci hlášení a přehlednou prezentaci agregovaných dat. Systém byl implementován ve složité heterogenní síti a úspěšně otestován. Jednoznačným výsledkem testu je výrazné zefektivnění správy sítě, a tím následné zvýšení její kvality a zabezpečení. Z posouzení správy a provozu sítě před a po implementaci systému vyplývá, že monitorování provozu pomocí systému tohoto typu vede k úsporám času potřebnému pro správu sítě i nákladů potřebných na řešení nečekaných havárií, které není možno bez podobného monitorovacího systému predikovat.

Celý systém je v současné chvíli implementován a přizpůsoben jedné konkrétní síti. Chtěl bych časem systém dopracovat do podoby, která by umožňovala jeho větší a obecnější nasazení. Ať už to bude formou systému šablon, který by bylo možné nasadit podle typu monitorovaných zařízení, nebo jiným způsobem. Cílem bude se maximálně vyrovnat s již zmiňovanou nesourodostí logových zpráv z různých prvků.

Dalším možným úkolem je řešení dalších způsobů, jak přenést informace z monitorovaných systémů na centrální server. U monitorovacího Controleye - Agenta jsem zmiňoval možnost rozšířit odesílání dat přes http protokol. U aktivních síťových prvků je možné naprogramovat vlastní syslogový server, který by vyhovoval přesně potřebám systému Controleye.

Výzvou také může být, nalezení řešení na komunikaci z centrálního panelu směrem k zařízení tak, aby nedošlo k narušení soukromí uživatelů systému a zároveň se ulehčila práce správcům. Zejména v případě potřeby oprav konfigurací jednotlivých klientských stanic se usnadní práce správce tím, že by nemusel k danému zařízení chodit. Ze svých osobních zkušeností s monitorováním notebooků, které nosí uživatelé vždy při sobě, vím, že oprava jejich konfigurací je značný problém.

Věřím, že nejen díky vlastnímu implementovanému systému a jeho přínosu, ale i díky uvedeným nápadům a postřehům, si myslím, že má diplomová práce může ukázat cestu příštím řešitelům podobných problémů a síťovým správcům, kteří by měli chuť se pouštět do obdobného díla. Dle mého názoru je největší přínos v poznání, že na nesourodých prvcích sítě se většinou bude hůře hledat společný průnik v seskupování informací zaznamenaných do logových protokolů (resp. souborů).

Pro mě osobně tato aplikace má velký přínos v samotné podstatě. Vznikl systém, který mě, jakožto správci systému, ulehčil práci při správě sítě. Byl vyvinut v rámci studijní aktivity, takže jeho cena je v nákladech nulová, pominu-li svůj investovaný čas. Současně s tím jsem ulehčil i situaci uživatelům sítě v mnoha pochůzkách a hledáních správce sítě, který by jim odstranil problém, o kterém správce do této chvíle netušil.

Proto přímý přínos z této diplomové práce nebudu mít jen já, ale i uživatelé takto monitorované sítě a doufám, že tato práce může také ukázat směr dalším případné diplomantům, či systémovým analytikům.

Seznam použitých zkratek

Zkratka	Anglický význam
IDS	**Intrusion detection system**
IPS	**Intrusion prevention system**
SNMP	**Simple Network Management Protocol**
WMI	**Windows Management Instrumentation**
MARS	**Monitoring, Analysis and Response System**
IP	**Internetový protokol**
HTTP	**Hypertext Transfer Protocol**
FTP	**File Transfer Protocol**
XML	**Extensible Markup Language**
MS-SQL	**Microsoft Server SQL**
MYSQL	**MySQL databases system**
ODBC	**Open Database Connectivity**
PHP	**Hypertext Preprocessor**
CPU	**Central Processing Unit**
VLAN	**Virtual Local Area Network**
ICMP	**Internet Control Message Protocol**
OID	**Object Identifier**
IPMI	**Intelligent Platform Management Interface**
TCP	**Transmission Control Protocol**
MAC	**Media Access Control**

Použitá literatura

Zabixový systém [online]. [cit. 2012-05-03]. Dostupné z: http://www.zabbix.org/

Systém Centreon [online]. [cit. 2012-05-03]. Dostupné z: http://www.centreon.com/

Česká wikipedeie [online]. [cit. 2012-05-03]. Dostupné z: http://cs.wikipedia.org/

GFI [online]. [cit. 2012-05-03]. Dostupné z: http://www.gfi.com/

Solarwinds [online]. [cit. 2012-05-03]. Dostupné z: http://www.kiwisyslog.com/

Seznam příloh

Příloha A: Adresářová struktura přiloženého CD

/Controleye-agent	Aplikace pros sběr dat
/Controleye-Agent-projekt	Zdrojové kódy aplikace pro sběr dat
/CtiMe.txt	Popis obsahu
/Controleye-Dashboard	Webová aplikace

Ověření technologie Traffic-Flow na platformě Mikrotik a NetFlow na platformě Cisco

Daniel Stříbný a Ondřej Pavlík

Abstrakt: Cílem tohoto díla byla dokumentace zprovoznění technologie Traffic-flow na platformě Microtik a NetFlow na platformě Cisco a jejich vzájemné porovnání.

Klíčová slova: Cisco, Mikrotik, NetFlow, Traffic-flow

Technologie pro vyhodnocování provozu

Technologie NetFlow a Traffic-flow

NetFlow je protokol od společnosti Cisco Systems tento protokol je otevřený, proto jej společnost Mikrotik použila do svých zařízení a díky tomuto zakomponování jednotného protokolu je možné mít na síti zařízení obou firem a vyhodnocovat získané údaje jedním programem. Obě tyto technologie slouží pro monitorování síťového provozu. V současné době pomocí tohoto protokolu mohou administrátoři díky těmto statistikám zdokonalovat síť tak aby nikde nedocházelo k většímu přetížení síťových prvků.

Architektura protokolu

Architektura celé služby se většinou skládá z X zařízení tvořících exportéry dat a jednoho bodu do kterého se tato data zasílají a v kterém se zpracovávají tzv. kolektor. Exportér je připojen k monitorované lince a analyzuje procházející pakety. Ze zachycených toků generuje statistiky. Ty pak shromažďuje na kolektoru. Kolektor je zařízení s velkou úložnou kapacitou obvykle nějaký server (pro menší sítě stačí PC), které sbírá statistiky (data) z většího počtu exportérů a ty ukládá na svých diskových kapacitách. Tyto data se dále vyhodnocují a s pomocí grafických nástrojů se z nich vytvářejí souhrnné přehledy.

Realizace výpočtu statistik na směrovačích

Architektura Cisco předpokládá na pozici exportérů směrovače, které vedle své hlavní činnosti provádějí také výpočet statistik. Tato architektura však trpí několika nevýhodami. Nevýhodou čílo jedna je vysoká pořizovací cena zařízení s dostatečným výkonem, které by zvládla současně dostatečně rychle směrovat síťový provoz a ještě vyhodnocovat statistiky. Proto většina směrovačů v dnešní době využívá vzorkování. Zachytávají se jen některé pakety (respektive každý n-tý).

Realizace výpočtu statistik na kolektorech

Aktuálnější a provozu atraktivnějším řešením se stává využití pasivních sond. Sondy ruší všechny nevýhody předchozí architektury, protože na rozdíl od směrovačů je lze připojit do libovolného bodu v síti a to tak aby neovlivňovaly přenosy v síti. Sondy procházející data pouze monitorují a nijak do nich nezasahují (proto pasivní sondy). Exportované statistiky jsou na kolektor odesílány vyhrazenou linkou a díky tomu jsou na monitorované lince zcela skryté (neovlivňují statistiky).

Struktura a verze Netflow a Traffic-flow

V Netlow packetu (zde příklad pro verzi 5) jsou obsaženy následující údaje:

- Číslo verze
- Sekvenční číslo
- Indexy vstupního a výstupního interfacu používané v SNMP
- Časové známky pro začátek a konec toku (v milisekundách od posledního startu zařízení)
- Počet bytů a packetů obsažených v toku
- Údaje z hlaviček 3 vrstvy:
- Zdrojová a cílová adresa
- Čísla zdrojového a cílového portu
- verze IP protokolu
- Type of Service (ToS)
- V případě TCP toku také TCP flags
- Informace o směrování na 3 vrtvě:
- IP adresa dalšího přeskoku
- Zdrojová a cílová maska

Cisco Netflow existuje v 10 verzích:

- V1 – první implementace, zastaralá, omezení na IPV4
- V2 – interní verze Cisco
- V3 – interní verze Cisco
- V4 – interní verze Cisco
- V5 – nejrozšířenější verze, dostupnám na routerech mnoha značek, omezena na IPV4
- V6 – dále nepodporovaná
- V7 – stejná jako V5, navíc pole s informací o zdrojovém routeru
- V8 – možnosti agregace, jinak stejná jako V5
- V9 – založena na šablonách, umí IPV6
- IPFIX (V10) - IETF standardizovaná V9 s rozšířeními

Mikrotik Traffic-flow podporuje Netflow protokol V1, V5 a V9. Doporučuje se používat samozřejmě nejvyšší dostupnou verzi (tedy V9).

... „Celou přílohu najdete v elektronické podobě na přiloženém CD disku.“